COMPUTATIONALLY EFFICIENT DISTRIBUTED MINIMUM WILCOXON NORM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

In

Communications and Signal Processing

By

RIPAN KUMAR SAHU

211EC4110

Department of Electronics and Communication Engineering
National Institute Of Technology
Rourkela

2013

COMPUTATIONALLY EFFICIENT DISTRIBUTED MINIMUM WILCOXON NORM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
In
Communications and Signal Processing

By

RIPAN KUMAR SAHU

211EC4110

Under the Guidance of

Prof. U.K.SAHOO

Department of Electronics and Communication Engineering
National Institute Of Technology
Rourkela
2013

National Institute Of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled, "COMPUTATIONALLY EFFICIENT DISTRIBUTED MINIMUM WILCOXON NORM" submitted by Mr. RIPAN KUMAR SAHU in partial fulfillment of the requirements for the award of Master of Technology Degree in Electronics & communication Engineering with specialization in "Communication and Signal Processing" at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date:

Prof. U. K. SAHOO

Department Of Electronics and Communication Engg.

National Institute of Technology

Rourkela-769008

ACKNOWLEDGEMENTS

This project is by far the most significant accomplishment in my life and it would be impossible without people who supported me and believed in me.

I would like to extend my gratitude and my sincere thanks to my honorable, esteemed supervisor Prof. U.K. SAHOO. He is not only a great lecturer with deeper vision but also and most importantly a kind person. I sincerely thank for his exemplary guidance and encouragement in my life. His trust and support inspired me in the most important moments of making right decisions and I am glad to work with him.

I want to thank all my teachers Prof. K. K. Mahapatra, Prof. S.K. Patra and Prof. S.K. Meher for providing a solid background for my studies and research thereafter. They have been great sources of inspiration to me and I thank them from the bottom of my heart.

I would like to thank all my friends and especially my classmates for all the thoughtful and mind stimulating discussions we had, which prompted us to think beyond the obvious. I've enjoyed their companionship so much during my stay at NIT, Rourkela.

I would like to thank all those who made my life in Rourkela an unforgettable and rewarding experience.

Finally I would like to thank my parents for raising me in a way to believe that I can achieve anything in life with hard work and dedication.

RIPAN KUMAR SAHU

ABSTRACT

In the fields related to digital signal processing and communication, as system identification, noise cancellation, channel equalization, and beam forming Adaptive filters play an important role. In practical applications, the computational complexity of an adaptive filter is an important consideration. As it describes system reliability, swiftness to real time environment least mean squares (LMS) algorithm is widely used because of its low computational complexity (O (N)) and simplicity in implementation. The least squares algorithms, having general form as recursive least squares (RLS), conjugate gradient (CG) and Euclidean direction search (EDS), can converge faster and have lower steady-state mean square error (MSE) than LMS. However, for their high computational complexity (O (N^2)) makes them unsuitable for many real-time applications. Therefore controlling of computational complexity is obtained by partial update (PU) method for adaptive filters. A partial update method is implemented to reduce the adaptive algorithm complexity by updating a fraction of the weight vector instead of the entire weight vector. An analysis of different PU adaptive filter algorithms is necessary, sufficient so meaningful. The deficient-length adaptive filter addresses a situation in system identification where the length of the estimated filter is shorter than the length of the actual unknown system. System is related to the partial update adaptive filter, but has distinct performance. It can be viewed as a PU adaptive filter, in that machine the deficient-length adaptive filter also updates part of the weight vector. However, it updates some part of the weight vector in every iteration. While the partial update adaptive filter updates a different part of the weight vector for each iteration.

An adaptive distributed estimation strategy is mathematically deprived from incremental gradient techniques. The prescribed scheme addresses the problem of distributed linear estimation in a cooperative fashion, concluding in a distributed algorithm that can respond in real time to changes in the environment. Every node is capable to communicate only with its immediate neighbor in order to exploit the spatial dimension while at the same time reducing the communications hurdle. A spatial-temporal energy conservation argument is used to evaluate the steady-state mean-square-error performance of the individual nodes across the adaptive distributed network. Computer MATLAB simulations illustrate the results.

Table of Contents

LIST OF FIGURE

ABBREVIATIONS

LMS	**Least Mean Square**
RLS	**Recursive Least Square**
P-LMS	**periodic LMS**
S-LMS	**sequential LMS**
SPU-LMS	**Stochastic Partial Update LMS**
MAX-PU LMS	**Maximum Partial Update LMS**
MSE	**Mean Square Error**
MSD	**Mean Square Deviation**
EMSE	**Excess Mean Square Error**
IMWN	**Incremental Minimum Wilcoxon Norm**
SS-IMWN	**Sign-Sign Incremental Minimum Wilcoxon Norm**
SR-IMWN	**Sign-Regressor Incremental Minimum Wilcoxon Norm**
IMSWN	**Incremental Minimum Sign Wilcoxon Norm**

Chapter 1

Introduction

1. CHAPTER 1::INTRODUCTION

1.1 MOTIVATIONS

Wireless Sensor Networks (WSNs) is networks composed of tiny embedded devices. Each device is capable of sensing, processing and communicating the local information. The networks can be made up of hundreds or thousands of devices that work together to communicate the information that they obtain. Distributed processing deals with the extraction of information from data collected at nodes that are distributed over a geographic area. As each node in a network of nodes could collect noisy observations related to a certain parameter or the phenomenon of interest. The nodes would then interact with their neighbors in a certain manner, as dictated by the network topology, in order to derive at an estimate of the parameter or phenomenon of interest. The objective is to arrive at an estimate that is as accurate as the one that would be obtained if each node had access to the information across the entire network. In comparison, in a traditional centralized solution, the nodes in the network would collect observations and send them to a central location for processing. The central processor unit would then perform the required estimation tasks and broadcast the result back to the individual nodes. This operation mode requires a powerful central processor, in addition to extensive amount of communication between the nodes and the processor [1]. In the distributed solution, the nodes rely solely on their local data and on interactions with their immediate neighbors. The amount of processing and communication is significantly reduced.

In a WSN each node is responsible for covering a particular area by sensing. The node then sends the result to a sink node that collects the data. Nodes are as usual used to relay the information, allowing the message to use multiple hops to reach the sink node. In order to process the information effectively, the network must have good coverage and the sink node must have good connectivity. Wireless Sensor Networks are frequently ad hoc, meaning that nodes can be added at any time and configure themselves to be part of the existing network. Any node can act as a relay to pass messages along the network. This works well for applications that add new sensors to replace those that have used up their battery life, or need to add more nodes for better coverage. Hence sensor placement needs to be done carefully considering the issues like coverage and connectivity.

Over the last decades adaptive digital filtering is a major area of research and has been applied in many contexts such as non-linear system identification, forecasting of time-series, beam forming channel linear prediction, equalization, line enhancer and noise cancellation. Adaptive digital filter self-adjusts its transfer function according to an optimizing algorithm to minimize the mean square between its output and that of an unknown system.

The Least Mean Square (LMS) algorithm is widely used because of its low computational complexity (O (N)) and simplicity in implementation. However, it is well known that the LMS has low convergence speed, especially for correlated input signals. The least squares algorithms as Recursive Least Squares (RLS), Conjugate Gradient (CG), and Euclidean Direction Search (EDS), can converge fast and have low steady-state mean square error (MSE). However, with high computational complexity (O (N^2)), these algorithms need expensive real-time resources, i.e., clock cycles, memory and power in a digital signal processor (DSP) or field-programmable gate array (FPGA). A well-known approach to controlling computational complexity is applying partial update (PU) method for adaptive filters. Partial updating of the LMS adaptive filter has been proposed to reduce computational costs and power consumption. A partial update adaptive filter reduces computational complexity by updating part of the coefficient vector instead of updating the entire vector or by updating part of the time. However, the partial update adaptive filters may converge faster than the full-update filters and achieve lower steady-state MSE in particular applications. In the literature, partial update methods have been applied to several adaptive filters, such as LMS, NLMS, RLS, Affine Projection (AP), Normalized Constant Modulus Algorithm (NCMA), etc. However, there are only a few analyses of these partial update adaptive filter algorithms. Many analyses are based upon partial update LMS and its variants. Only a few papers have addressed partial update RLS and AP.

In a WSN each node is responsible for covering a particular area by sensing. The node then sends the result to a sink node that collects the data. Nodes are used to relay the information, allowing the message to use multiple hops to reach the sink node. In order to process the information effectively, the network must have good coverage and the sink node must have good connectivity. Wireless Sensor Networks are frequently ad hoc, meaning that nodes can be added at any time and configure themselves to be part of the existing network. Any node can act as a relay to pass messages along the network. This works well for applications that add new sensors to replace those that have used up their battery life, or need to add more nodes for better coverage. Hence sensor placement needs to be done carefully considering the issues like coverage and connectivity.

1.2 PROBLEM STATEMENT

Adaptive digital filtering self-adjusts its transfer function to get an optimal model for the unknown system based on some function of error based on the output of the adaptive filter and the unknown system. To get an optimal model of the unknown system it depends on the structure, adaptive algorithm and the nature of the input signal. System Identification estimates models of dynamic systems by observing their input output response when it is difficult to obtain the mathematical model of the system.

Mathematical analysis has also been extended to the existing PU adaptive filter algorithms. This work has analyzed the convergence conditions, steady-state performance, and tracking performance. The theoretical performance is validated by computer simulations. The performance is compared between the original adaptive filter algorithms and different partial-update methods. Since a specific PU method in one adaptive filter algorithm which achieves good performance may not perform well in another adaptive filter algorithm, the performance of one PU method for different adaptive filter algorithms is also compared. Computational complexity is calculated for each partial-update method and each adaptive filter algorithm.

In wireless sensor network the fusion center provides a central point to estimate parameters for optimization. Energy efficiency i.e. low power consumption, low latency, high estimation accuracy and fast convergence are important goals in estimation algorithms in sensor network. Depending on application and the resources, many algorithms are developed to solve parameter estimation problem. One approach is the centralized approach in which the most information to be present when making inference. However, the main drawback is the drainage of energy resources to transmit all observation to fusion center at every iteration. So this is wasting energy at idle time interval. Hence there was a need to find an approach that avoids the fusion center all together and allows the sensors to collaboratively make inference. This approach is called as the distributed scheme. Distributed computation of algorithms among sensors reduces energy consumption of the overall network, by tradeoff between communication cost and computational cost. In order to make the inference procedure robust to nodal failure and impulsive noise, robust estimation procedure should be used. Optimization of sensor locations in a network is essential to provide communication for a longer duration. In most cases sensor placement needs to be done in hostile areas without human involvement, e.g. by air deployment. The aircraft carrying the sensors has a limited payload, so it is impracticable to randomly drop thousands of sensors over the ROI. Thus, the objective must be performed with a fixed number of sensors. The air deployment may introduce uncertainty in the final sensor positions. These limitations motivate the establishment of a planning system that optimizes the WSN deployment process.

In the field of signal processing and communication Adaptive Filtering has a tremendous application such as non-linear system identification, forecasting of time-series, linear prediction, channel equalization, and noise cancellation. Adaptive digital filtering self-adjusts its transfer function to get an optimal model for the unknown system based on some function of error based on the output of the adaptive filter and the unknown system. To get an optimal model of the unknown system it depends on the structure, adaptive algorithm strategy and the nature of input signal.

System Identification estimates models of dynamic systems by observing their input output response when it is difficult obtain the mathematical model of the system.

DSP-based equalizer systems have become ubiquitous in many diverse applications including voice, data, and video communications via various transmission media. Typical applications range from acoustic echo cancellers for full-duplex speakerphones to video de-ghosting systems for terrestrial television broadcasts to signal conditioners for wire line modems and wireless telephony. The effect of an equalization system is to compensate for transmission-channel impairments such as frequency-dependent phase and amplitude distortion. Rather for correcting for channel frequency-response ambiguity, cancel the effects of Multipath signal and to reduce the inter-symbol interference. So, construction of Equalizer to work for the above specifications is always a challenge and an active field of research.

On-line system identification or identification of complex systems is a major area of research from last several years. To abstract a new solution to some long standing necessities of automatic control and to work with more and more complex system to satisfy stricter design criteria and to fulfill previous points with less and less a priori knowledge of the unknown system. In this context a great effort is being made within the system identification towards the development of nonlinear models of real processes with less no of mathematical complexity, less no of input sample, faster matching and better convergence. This has been verified by MATLAB simulation version 2010.

1.3 THESIS LAYOUT

In Chapter2, the Adaptive Filter and System Identification problem are discussed in brief and an Adaptive Model for System Identification problem is given. Furthermore the nonlinear issues in the System Identification problems are discussed. This chapter is focused on the basics of the robust system.

In chapter 3, an introduction to wireless sensor network and a new type of distributed cooperative strategy based on incremental technique. The uncorrelated but identical node is trained with an LMS algorithm for local parameter estimation by sharing immediate neighbor's system parameter. Adaptive modeling and system identification problem is defined for linear and nonlinear plants. The conventional LMS algorithm and other gradient based algorithm for the FIR system are derived. Nonlinearity problems are discussed briefly and various methods are proposed for its solution. The steady state analysis of incremental LMS is shown both theoretically and using simulation.

Chapter 4 describes the partial updating technique in the LMS strategy algorithm. Many other application areas of LMS include interference cancellation echo cancellation, space time modulation and coding, signal copy in surveillance, and wireless communications. Although there exist algorithms with swifter convergence rates like RLS, LMS is popular because of its ease of implementation and low computational costs. Partial updating of the LMS adaptive filter has been proposed to reduce computational costs and power consumption, which is quite attractive in the area of mobile computing and communications. Many mobile communication devices have applications like channel equalization and echo cancellation that require the adaptive filter to have a very large number of coefficients. Updating the entire coefficient vector is costly in terms of power, memory, and computation and is sometimes impractical for mobile units.

In chapter 5 diffusion strategy in a distributed network is analyzed. The problem of distributed estimation a set of nodes is required to collectively estimate some parameter of interest from noisy measurements. The problem is useful in several contexts including wireless and sensor networks, where robustness, scalability and low power consumption are desirable features. Diffusion cooperation algorithm schemes have been shown to provide good performance, robustness to noise at node and link failure, and are amenable to distributed implementations. In this work focus is on diffusion-based adaptive solutions of the LMS type. The motivation and propose new versions of the diffusion LMS algorithm that outperform previous solutions. So the performance and convergence analysis of the proposed algorithms together with simulation results is compared with existing techniques. We also discuss optimization schemes to design the diffusion LMS weights.

Chapter 2

Distribute Adaptive System Identification

2. Chapter 2 :: Distribute Adaptive System Identification

2.1 Basic Adaptive Filter Models

System identification is used to estimate an unknown linear or nonlinear system with digital signal processing [2] [3]. System identification processed through the steps as given below:

a. Experimental design: Its purpose is to obtain good experimental data and it includes the Choice of the measured variables and of the character of the input signals.
b. Selection of model structure: A suitable model structure is chosen using prior knowledge and trial and error.
c. The choice of the criterion to fit: A suitable cost function is chosen, which reflects how well the model fits the experimental data.
d. Parameter estimation: An optimization problem is solved to obtain the numerical values in the model parameters.
e. Model validation: The model is tested in order to reveal any inadequacies.

System identification has a constraint of finding an appropriate model structure for the system. Mounting a model within a given structure (parameter estimation) is at utmost a generic hazard. In other words, one should utilize prior knowledge and physical insight about the system when choosing the model structure. There are three levels of prior knowledge, which have been color coded as follows.

White Box models: In case when a model is perfectly known; it has been possible to construct it entirely from prior knowledge and physical insight.

Grey Box models: In case when a model is slightly known, several parameters have concluded from observed data.

Black Box models: No physical insight is available or used, but the chosen model structure belongs to the families that are known to have good flexibility and have been "successful in the past".

The adaptive systems have the following characteristics

1. They can automatically adapt (self-optimize) in the face of changing (non-stationary) environments and changing system requirements.
2. They can extrapolate a model of behavior to deal with new situations after trained on a finite and often small number of training signals and patterns.

3. They can repair themselves to a limited extent.
4. They can be described as nonlinear systems [2] with time varying parameters.
5. They can be trained to perform specific filtering and decision making tasks.

2.2 System Identification Model

A block diagram for system identification is given in Figure 2-1. An input is applied to the unknown system and the adaptive filter simultaneously. Usually a noise will be added at the output of the unknown system [4] [5]. If the unknown system does not change with time, then it is a time-invariant system. If the unknown system changes with time, then it is a time-varying system.

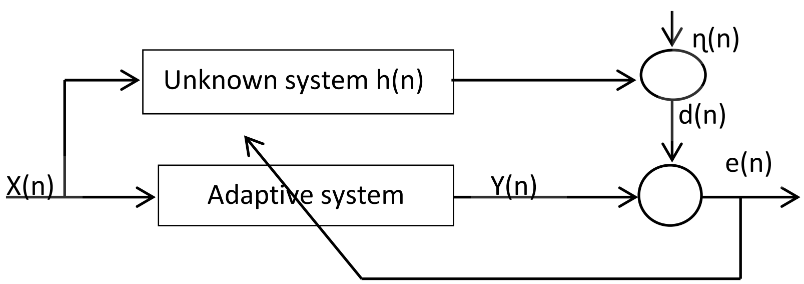


Figure 2-1 general adaptive filter model

The system identification model can be presented as:

$$d(n) = x^T(n)w(n) + \eta(n) \qquad \text{2-1}$$

Where d (n) is the desired signal, x (n) = [x(n), x(n - 1), ..., x(n - N + 1)]^T is the input data vector of the unknown system

$$W^0(n) = [w_1^0, w_2^0, w_3^0, w_4^0, \ldots, w_N^0] \qquad \text{2-2}$$

The impulse response vector of the unknown system, and v (n) is zero-mean white noise, which is independent of any other signals.

Considering coefficient vector of adaptive system as W, The estimated signal y(n) is defined as

$$y(n) = x^T(n)w(n) \qquad \text{2-3}$$

The output signal error is defined as

$$e(n) = d(n) - x^T(n)w(n) \quad \textbf{2-4}$$

This error signal is feedback to the adaptive system so as to update the system function iteratively by update strategy.

2.3 Filter Structure

In general, any system with a finite number of parameters that affect how u (n) is computed from x (n) could be used for the adaptive filter in Figure 2-1. Define the parameter or coefficient vector W (n):

$$W(n) = [w_0(n)\ \ w_1(n)\ w_2(n) \dots w_{L-1}(n)]^T \quad \textbf{2-5}$$

Where $\{w_i\ (n)\}$, $0 < i < L - 1$ are the L parameters of the system at time n. The general input-output relationship for the adaptive filter can be defined as:

$$y(n) = \sum_{k=1}^{M} a_k y(n-k) + \sum_{l=0}^{N} b_l x(n-l) \quad \textbf{2-6}$$

Where, M and N are positive integers. As from the equation it is assumed that the output is linearly or nonlinearly dependent on the input. Therefore the solution of differential equation relates to system causality.

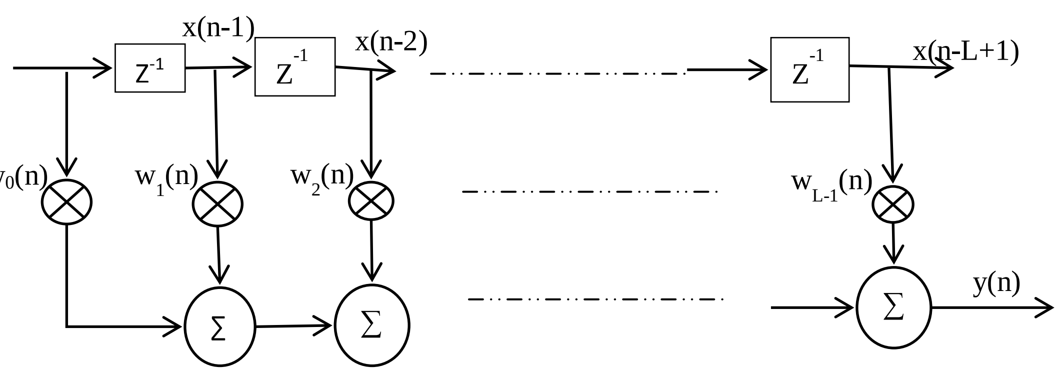


Figure 2-2 STRUCTURE OF FIR FILTER TAP DELAY LINE CONFIGURATION

Although Equation (2.3) is the most general description of an adaptive filter structure, we are interested in determining the best linear relationship between the input and desired response signals for many problems. This relationship typically is being either finite-impulse-response (FIR) or infinite-impulse-response (IIR) filter. Figure2.3 shows the structure of a direct-form FIR filter, also known as a taped-delay-line or transversal filter, where z^{-1} denotes the unit delay element and each w_i (n) is a multiplicative gain within the system. In this case, the parameters in w(n) correspond to the impulse response values of the filter at time n. We can write the output signal y(n) as:

$$\boldsymbol{y(n) = \sum_{i=0}^{L-1} w_i(n)x(n-i) = w^T(n).x(n)} \qquad \textbf{2-7}$$

As $X(n)=[x(n)\ x(n-1)\ x(n-2)\ x(n-L+1)]^T$ denotes input regression vector.

As per system requirement it is noted from an equation that L multipliers and L-1 adders are necessary to perform the operation. These computations are easily performed by a processor or circuit so long as L is not too large and the sampling period. It also requires a total of 2L memory locations to store the L input signal samples and the L coefficient values, respectively. The adaption is of two types

I. Open loop adaption: it invokes measurement of input or ambient characteristics by applying gathered information to a formula or to a computational algorithm using generated results to set the adjustments of the adaptive system. This adaption does not depend on output signal.

II. Closed loop adaption: on the contrary involving output in determining the result is quite adequate. It involves automatic experimentation with these adjustment and knowledge of their outcome in order to optimize a measured system performance. It is also referred as performance feedback.

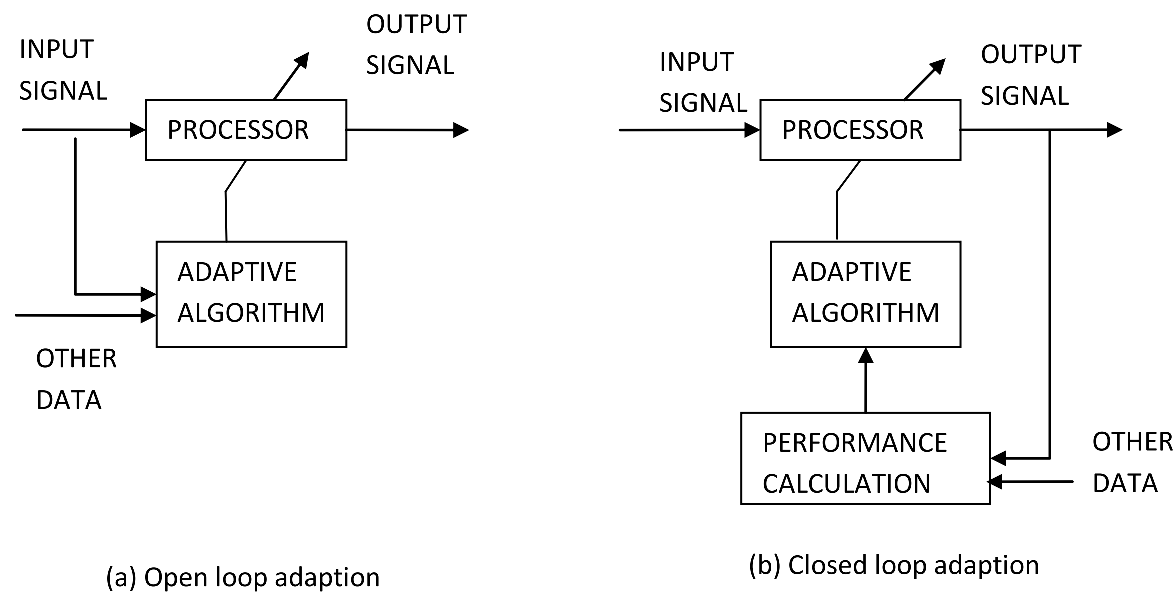


Figure 2-3 (a) open loop adaption (b) closed loop adaption

2.4 Adaptive Algorithms

For minimizing a cost function by adjusting parameters of adaptive filter there are numerous methods are optimized. in current session many general forms of adaptive filer are considered with simple derivation of the LMS adaptive algorithm. FIR system [6] is more popular that to IIR filter because

1. **FIR filter is more stable than IIR one.**
2. **Adjustment algorithm of FIR filter coefficient is easier.**

2.4.1 Adaptive FIR Algorithm

The conventional form of an adaptive FIR filtering algorithm is

$$\boldsymbol{w(n+1) = w(n) + \mu(n) G\big(e(n), X(n), \phi(n)\big)} \qquad \textbf{2-8}$$

where G(-) is a particular vector-valued nonlinear function, μ(n) is a step size parameter, *e(n)* and *X(n)* are the error signal and input signal vector, respectively, and Φ(n) is a vector for storing pertinent information about the characteristics of the input and error signals and the coefficients at previous time instants. In the simplest algorithms, Φ(n) is not used, and the only information needed to adjust the coefficients at time *n* are the error signal, input signal vector, and step size. The step size is so called because it determines the magnitude of the change or "step" that is taken by the algorithm in iteratively determining a useful coefficient vector. Much research effort has been spent characterizing the role that μ(n)plays in the performance of adaptive filters in terms of the statistical or frequency characteristics of the input and desired response signals. Often, success or failure of an adaptive filtering application depends on how the value of μ(n) is chosen or calculated to obtain the best performance from the adaptive filter.

2.4.2 Cost Function

The convergence of cost function leads to the solution of the equation. Dependence of cost function on adaptive algorithm yields its performance surface **[5]**. This is the convergence analysis plot. Generally the performance surface is a graph of parameters of system vs. MSE defined as by the following mean square error estimation equation given below

$$\boldsymbol{J_{MSE}(n) = 0.5 \int_{-\infty}^{\infty} e^2(n) p_n\big(e(n)\big) d[e(n)] = 0.5\, E[e^2(n)]} \qquad \textbf{2-9}$$

Where

$P_n(e(n))$: the probability density function of the error at time *n.*

E[]:expectation integral.

It enables us to determine both the optimum coefficient values given knowledge of the statistics of d(n) and x(n) as well as a simple iterative procedure for adjusting the parameters of an FIR filter. J_{MSE} is a smooth function of each of the parameters in W(n), such that it is differentiable with respect to each of the parameters in W(n).

2.5 FIR Filter Weiner Solution

To determine optimum solution in performance surface, minimize $J_{MSE}(n)$ if the statistics of the input and desired response signals are known. To determine W_{MSE} (n) we note that the function J_{MSE}(n) in (2.10) is quadratic in the parameters $\{w_i(n)\}$, and the function is also differentiable. Thus, utilizing the result from optimization theory that states that the derivatives of a smooth cost function with respect to each of the parameters is zero at a minimizing point on the cost function error surface. Thus, $W_{MSE}(n)$ can be found from the solution to the system of equations.

$$\frac{\partial J_{MSE}(n)}{\partial w_i(n)} = 0 \dots\dots\dots 0 \leq i \leq L-1 \qquad \textbf{2-10}$$

$$= -E\left\{e(n) \times \frac{\partial y(n)}{\partial w_i(n)}\right\}$$

$$= -E\{e(n) \times x(n-i)\}$$

$$= -E\{d(N)x(n-i)\} - \sum_{j=0}^{L-1} E\{x(n-i)x(n-j)\}\, w_j(n)$$

Where

$$R_{xx} = E\{X(n)X^T(n)\}$$

$$P_{xx} = E\{d(n)X^T(n)\} \qquad \textbf{2-11}$$

Respectively

$$R_{xx}(n)W_{MSE}(n) - P_{dx}(n) = 0$$

$$W_{MSE}(n) = R_{xx}^{-1}(n)P_{dx}(n) \qquad \textbf{2-12}$$

Steepest descent is a celebrated optimization procedure for minimizing the value of a cost function *J(n)* with respect to a set of adjustable parameters *W(n)*.

$$w_i(n+1) = w_i(n) - \mu(n)\frac{\partial J(n)}{\partial w_i(n)} \qquad \textbf{2-13}$$

$$w(n+1) = w(n) + \mu(n)\big(P_{dx}(n) - R_{xx}(n)w(n)\big) \qquad \textbf{2-14}$$

$$w(n+1) = w(n) + \mu(n)e(n)X(n) \qquad \textbf{2-15}$$

2.6 Distributed Sensor Network

Networks consisting of nodes collecting data over a geographical area are envisioned to make a dramatic impact on a number of applications such as precision agriculture, disaster relief management, radar and acoustic source localization. In these applications, each node has some computational power, is adept to send data to a subset of the network nodes [7], and tries to estimate a parameter of interest [8]. Therefore, there is a great deal of effort in devising algorithms that are able to improve the estimate of the parameter of interest in every node with this information exchange between nodes [9]. More precisely, in mathematical terms, each node should optimize a cost function that depends on the information available on the network.

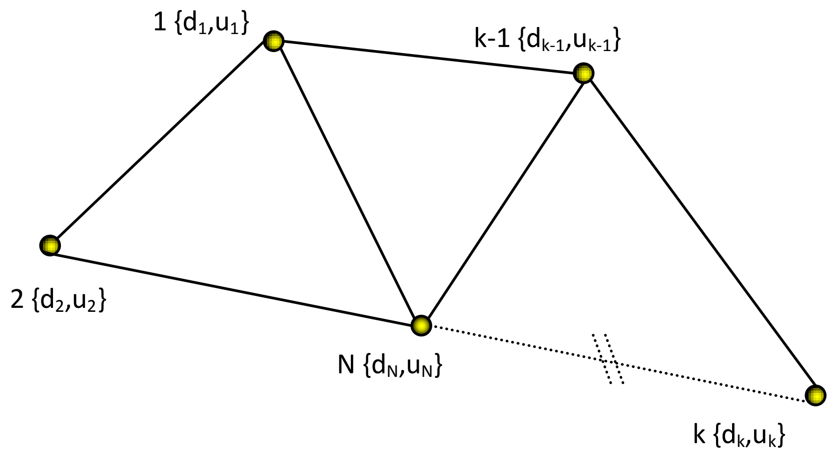


Figure 2-4 Distributed network with N active nodes accessing space-time data

The Least Mean Square (LMS) algorithm, introduced by Widrow and Hoff in 1959 [12] is an adaptive algorithm, which uses a gradient-based method of steepest decent [10]. The LMS algorithm uses the estimates of the gradient vector of the available data. LMS incorporates an iterative procedure that makes successive corrections to the weight vector in the direction of the negative of the gradient vector which eventually leads to the minimum mean square error. Compared to the other algorithms LMS algorithm is relatively simple; it does not require correlation function calculation nor does it require matrix inversions.

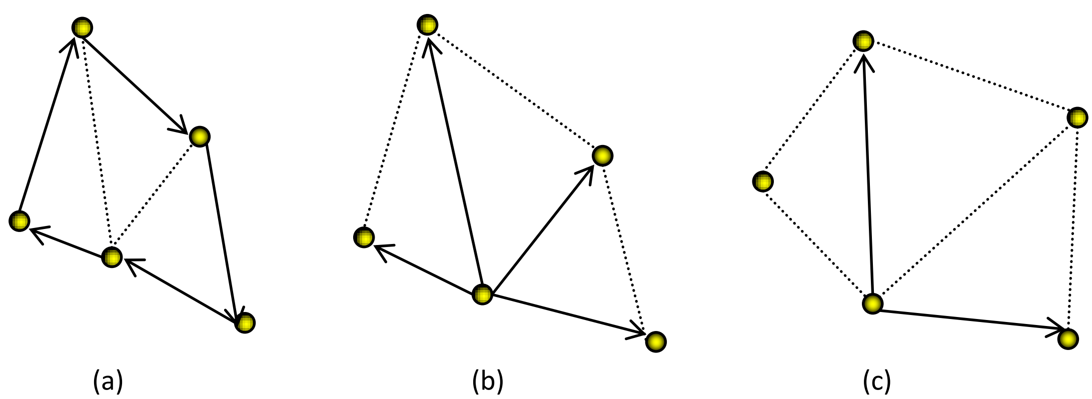


Figure 2-5 Three modes of operation (a) incremental; (b) diffusion; (c) probabilistic diffusion

As an efficient high durable system the power consumption thus computational complexity load should be controlled in adaptive filter implementations. Partial updating of LMS filter coefficients is one of durable methodology in this power maintenance scheme. The problem of distributed estimation, a set of node is required to collectively estimate some parameter of interest from noisy measurement. The problem is useful in several contexts including wireless and sensor networks, where robustness, scalability and low power consumption for longer operating life are desirable features. In the gradual approximation selection of filter coefficient are upgraded in regular fashion.

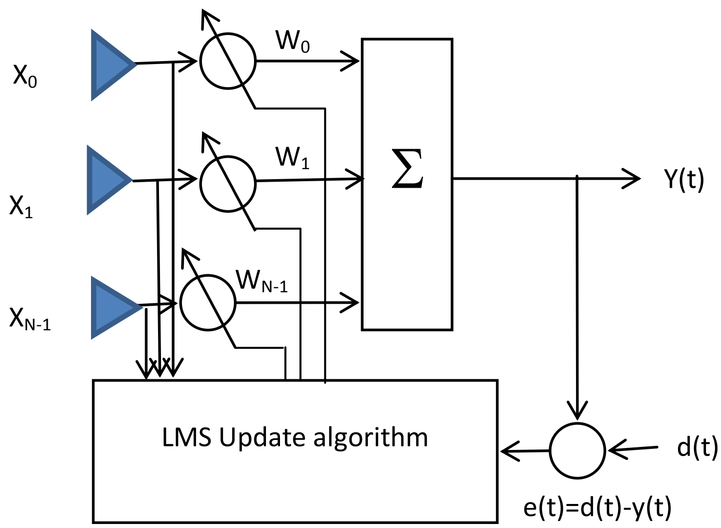
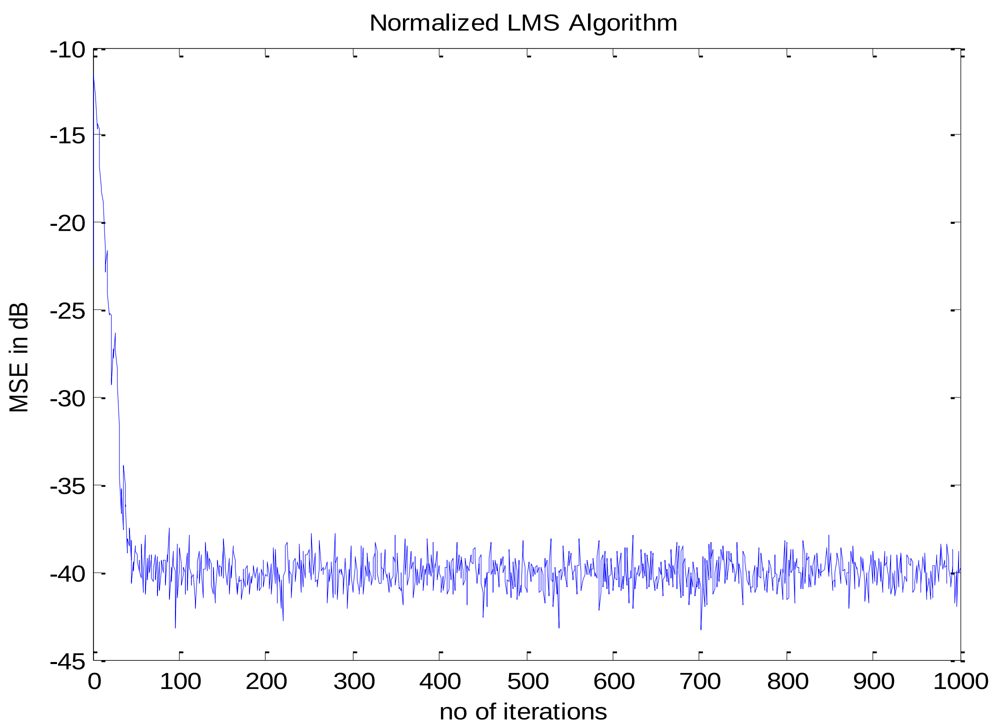


Figure 2-6 LMS adaptive beam forming network

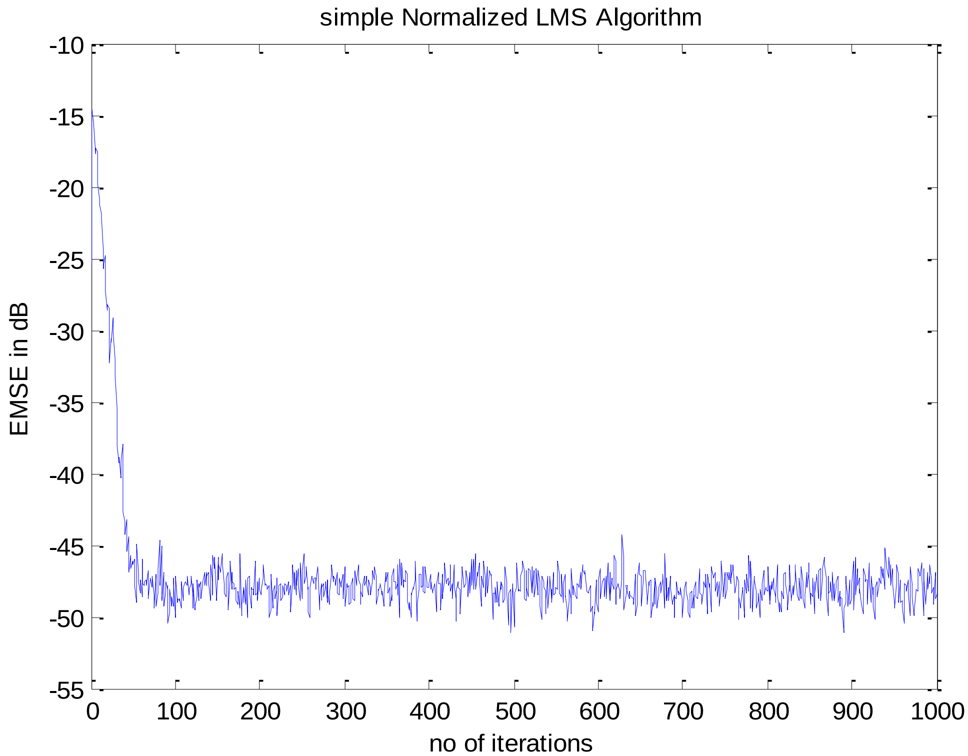
For real-valued data which can be extended for the analysis of complex valued data. Small bold letters are used to denote vectors, e.g., w denotes the vector and capital bold letter e.g. W denotes the matrix. The symbol super-script T denotes transposition of vector. The notation $\| w \|^2$ denotes the squared Euclidean norm of a vector$\| w \|^2 = w^T \times w$. Similarly $\| w \|_\Sigma^2$ denotes the weighted-squared Euclidean norm$\| w \|_\Sigma^2 = w^T \times \Sigma\, w$. All vectors are column vector except for the input data vector denoted by u_i, which is taken as a row vector. The time instant is placed as a subscript for vectors and between the parentheses for scalars, e.g. W_i and e(i).

2.7 Simulation Result

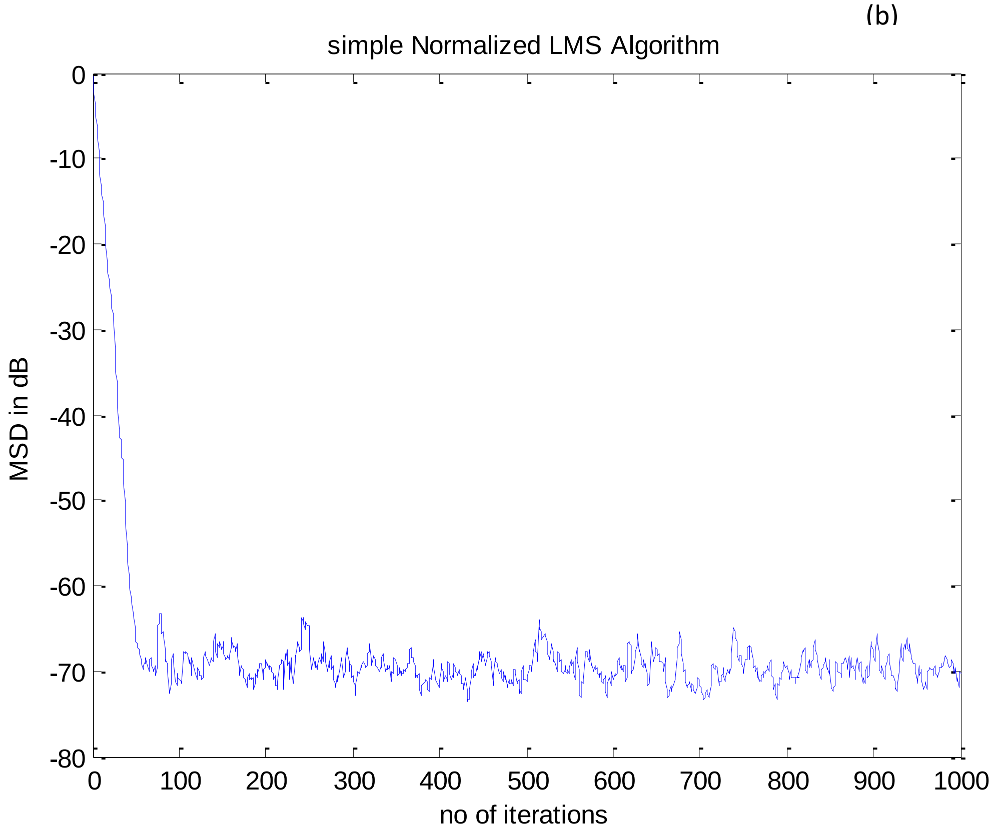
The performance of LMS algorithm is tested for both linear and nonlinear systems. For identification purpose a tap delay filter with three taps is used. The parameter of the linear part of the plant is h(n)= [0.26 0.93 0.26]. For the simulation the initial parameters of the model is taken as zeros. Gaussian noise of signal to noise ratio (SNR) 30dB was added which accounts for measurement noise. The input to the plant was taken from a uniformly distributed random signal over the interval [-0.5, 0.5] .The adaptation is continued for 2000 iterations which is ensemble over 50 iterations. After training filter weights remain fixed. For testing new 20 samples are generated and pass through the plant as well as model. The mean square error (MSE) and responses are plotted for the linear and nonlinear systems.



(a)



(b)



(c)

Figure 2-7 plot of NLMS algorithm (a)MSE (b)EMSE (c)MSD

2.8 Conclusion

Application of adaptive filter and two types of modeling is described in this chapter. System identification deals with direct modeling. The LMS algorithm is used for system identification purpose because of its simplicity. From Fig (2.7) to (2.11) it is observed that for linear system LMS algorithm based model gives best result. As the nonlinearity associated with the system goes on increasing the LMS based model response deviates from the actual response. Taking different types of nonlinearity the MSE and responses are plotted. From Fig.2.11 it is seen that the actual response and the LMS based model response do not match anywhere. From this a conclusion can be drawn that LMS based models are best for linear systems.

Chapter 3

Incremental Minimum Wilcoxon Norm

3. CHAPTER 3 INCREMENTAL MINIMUM WILCOXON NORM

3.1 Introduction

In modern communication distributed networks linking PCs, cell phones, laptops, sensors and actuators will form the backbone of future data, communication and control networks. Applications will be sensor networks to precision agriculture, environment monitoring and target localization [10], disaster relief management, smart spaces, as well as medical applications. In all these cases, the distribution of the nodes in the field yields spatial diversity, which can exploited alongside the temporal dimension in order to enhance the robustness of the processing tasks and improve the probability of signal and event detection in ambient. Collaborative signal processing has been advocated as a way to achieve the efficient fusion of information. Regardless of the cooperative technique adopted, it is an accepted fact nowadays that distributed processing will need to be both *adaptive* and *cooperative*. This is because not only the environmental conditions vary with time and space, but the network topology may vary.

The information from data collected at nodes that are distributed over a geographic area, are processed on processing network [11]. Each node in a network of nodes could collect noisy observations related to a certain parameter or the phenomenon of interest. The nodes would then interact with their neighbor's parameter in a certain manner, as programmed by the network topology, in order to arrive at an estimate of the parameter or phenomenon of interest. The objective is to arrive at an estimate that is to be obtained if each node had access to the information across the entire network. In comparison, in a traditional centralized solution, the nodes in the network would collect observations and send them to a central location for processing. The centralized processor would then perform the required estimation tasks and broadcast [12] the result back to the individual nodes. This mode of operation requires a powerful central processor, in addition to extensive amounts of communication between the nodes and the processor so consuming an enormous amount of power. In the distributed solution, contrary to incremental the nodes rely solely on their local data [13] and on interactions with their immediate neighbors. The amount of processing task and communications hazards is significantly reduced [14] [15].

The distributed network links cell phones, PCs, sensors, laptops and actuators forming the backbone of future data communication and control networks. Applications are range from sensor networks to precision agriculture, environment monitoring, smart spaces, disaster relief management, target localization as well as medical applications [10]. In all these cases, the distribution of the nodes in the field yields a spatial diversity, which can be exploited alongside the temporal dimension in order to enhance the robustness of the processing tasks and improve the probability of signal and event detection [16] [17].

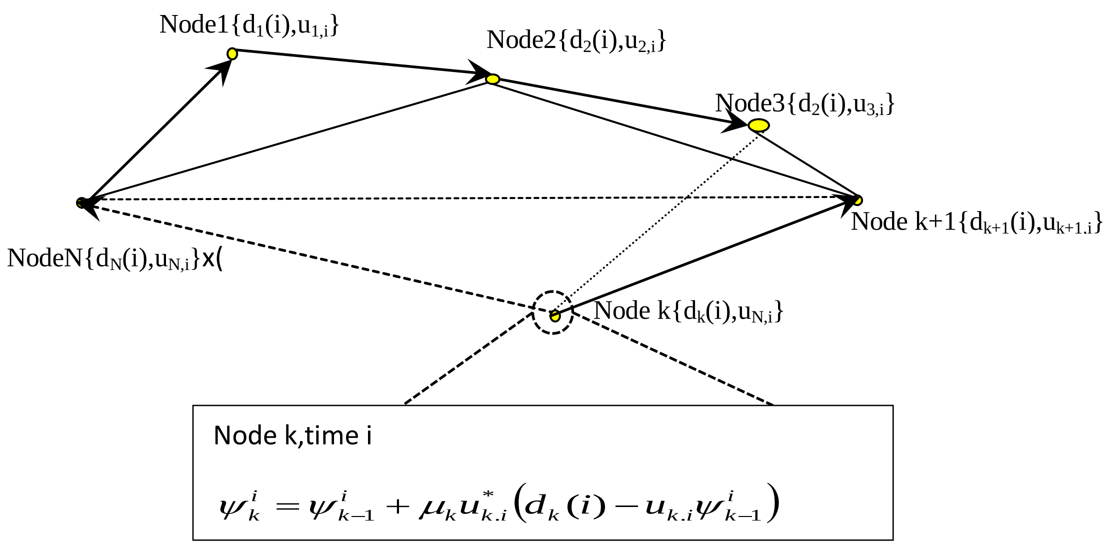


Figure 3-1 A distributed network with N nodes and the incremental algorithm path

3.2 Performance Analysis

In a network configuration containing N node can be updated by immediate neighbor node parameter. If sensor measures noisy vector from ambient such as local temperature, wind speed, humidity etc.at iteration 'i' node 'k' access random data$\{d_k, \mathbf{u_k}\}$,k=1,2,3,4,…..N. Where $d_k(i)$ is scalar measurement to be achieved, $\mathbf{u_{k,i}}$ is 1×M regressor vector.

Considering ψ_k^i denote a local estimate of optimum weight w0 at node k at time instant 'i'. If the node k has access to only to the neighbor parameter ψ_{k-1}^i, which is an estimate of w0 at its immediate neighbor node $k-1$ in the define topology. If at each time instant 'i' we start with the initial condition $\psi_0^i = w_{k-1}^i$ at node 1 (i.e., with the current global estimate w_{i-1} for w_0),and iterate goes on cyclically across the nodes then, at the end of the procedure the local estimate at node N will be assigned to w_i i.e. $w_i = \psi_N^i$. Now the distributed incremental LMS algorithm [4] is defined as follow as for each time i≥0, repeat:

$$u_{k,i} = [u_k(i), u_k(i-1), u_k(i-2), u_k(i-3), \ldots\ldots, u_k(i-M+1)].$$
$$\Psi_0^i = w_{i-1}$$
$$\Psi_0^i = \Psi_{k-1}^i + \mu_k u_{k,i}^*(d_k(i) - u_{k,i}\Psi_{k-1}^i) \qquad \textbf{3-1}$$
$$w_i = \Psi_N^i$$

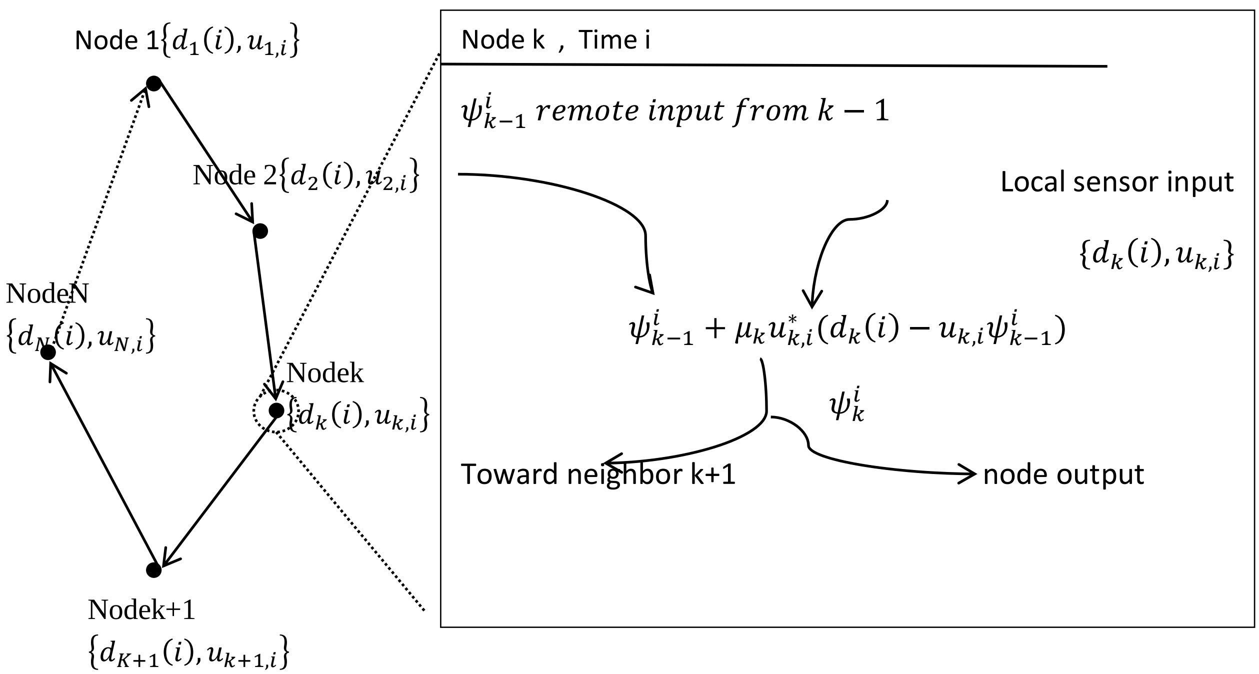


Figure 3-2 data processing proposed adaptive distributed structure

3.3 Data Modeling

The desired unknown vector w^0 relates to $\{d_k(i), U_{k,i}\}$ as

$$\boldsymbol{d_k(i) = u_{k,i}w^0 + v_k(i)} \qquad \textbf{3-2}$$

Where $v_k(i)$ is a white Gaussian noise with variance $\sigma_{v,k}^2$ and independent of $\{d_k(i), u_{k,i}\}$. If the input data $u_k(i)$ to the nodes are spatially and temporally independent. The local error signals at each node k are defined as

$$\bar{\psi}_k^i \quad = w^0 - \psi_k^i \qquad \textit{weight error at iteration i}$$

$$e_{a,k}(i) = u_{k,i}\bar{\psi}_{k-1}^i \qquad \textit{apriori error}$$
$$e_{p,k}(i) = u_{k,i}\bar{\psi}_k^i \qquad \textit{a posteriori error}$$
$$e_k(i) = d_k(i) - u_{k,i}\bar{\psi}_{k-1}^i \qquad \textit{output error} \qquad \textbf{3-3}$$

This output error e_k (i) estimate convergence with respect to the d_k (i) utilizing locally available information. as per the definition of $e_{a,k}$ (i) and data modelling equation

$$E\|e_k(i)\|^2 = E\|e_{a,k}(i)\|^2 + E\|v_k(i)\|^2 = E\|e_{a,k}(i)\|^2 + \sigma_{v,k}^2 \qquad \textbf{3-4}$$
$$\eta_k = E\|\bar{\psi}_{k-1}^{\infty}(i)\|^2 \qquad MSD$$
$$\zeta_k = E|e_{a,k}(\infty)|^2 \qquad EMSE$$
$$\xi_k = E|e_k(\infty)|^2 = \zeta_k + \sigma_{v,k}^2 \qquad MSE$$

The steady-state values of the variances like mean-square error (MSE), excess-mean-square error(EMSE) and mean-square deviation(MSD) for every node which are the measures of performance of the filter.where a weighted norm operator of a vector defined as$\|x\|_\Sigma^2 = x^2\Sigma x$ where ∑(≥0) is a Hermitian positive definite matrix.

$$e_{a,k}^{\Sigma}(i) = u_{k,i}\Sigma\bar{\psi}_{k-1}^i$$
$$e_{p,k}^{\Sigma}(i) = u_{k,i}\Sigma\bar{\psi}_k^i$$
$$e_{a,k}^{I}(i) = u_{k,i}\Sigma\bar{\psi}_{k-1}^i$$
$$e_{p,k}^{I}(i) = u_{k,i}\Sigma\bar{\psi}_k^i \qquad \textbf{3-5}$$

The incremental LMS can be written as

$$\psi_k^i = \psi_{k-1}^i + \mu_k u_{k,i}^* e_k(i) \qquad \textbf{3-6}$$

Subtracting w^0 from both sides,

$$\bar{\psi}_k^i = \bar{\psi}_{k-1}^i + \mu_k u_{k,i}^* e_k(i) \qquad \textbf{3-7}$$

Relation between various error terms $e_{a,k}^{\Sigma}(i), e_{p,k}^{\Sigma}(i), e_k(i)$ is obtained by pre-multiplying both side by $u_{k,i}\Sigma$ as

$$u_{k,i}\Sigma\bar{\psi}_k^i = u_{k,i}\bar{\psi}_{k-1}^i + \mu_k\|u_{k,i}^*\|_\Sigma^2 e_k(i)$$

$$e_{p,k}^{\Sigma}(i) = e_{a,k}^{\Sigma}(i) - \mu_k\|u_{k,i}^*\|_\Sigma^2 e_k(i)$$

$$e_k(i) = \frac{1}{\mu_k}\frac{e_{a,k}^{\Sigma}(i) - e_{p,k}^{\Sigma}(i)}{\|u_{k,i}\|_\Sigma^2} \qquad \textbf{3-8}$$

3.4 Statistical Regression Analysis

Wilcoxon norm is a robust norm used for statistical regression analysis. In Wilcoxon learning machine are designed for robustness against the outlier present in the desired. The convergence speed [18] of sign-regressor LMS and sign-sign LMS are faster than the LMS but it's performance decreases with respect to LMS. To define the Wilcoxon norm [19] a score function is required. The score function is $\varphi[0\ \ 1] \rightarrow \mathfrak{R}$ which is non-decreasing and bounded.

$$\int \boldsymbol{\varphi^2(u)du} < \infty \qquad \textbf{3-9}$$

The score value is $a(i) = \varphi(i/l + 1)$,'i' is fixed positive integer. The pseudo norm can be formulated by

$$\|\boldsymbol{v}\|_{\boldsymbol{w}} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{l}} \boldsymbol{a}(\boldsymbol{R}(\boldsymbol{v_i}))\, \boldsymbol{v_i} \qquad \textbf{3-10}$$

Where $V = [v_1\ v_2\ v_3\ \ldots.\ v_l]$ is size of vector. R(v_l) is rank of v_i.$a(i) = \varphi(R(v_i\ )/(l + 1))$

$$\boldsymbol{\varphi(u)} = \sqrt{\boldsymbol{12}}\ (\boldsymbol{u} - \boldsymbol{0.5})$$

For Wilcoxon norm score function can be modified to:

$$\boldsymbol{\varphi(u)} = \boldsymbol{sign}(\boldsymbol{u} - \boldsymbol{0.5})$$

3.4.1 Problem Formulation

The system model can be defined as:

$\mathrm{d_i} = \mathrm{u_i^T w + e_i + v_i}$, where $\mathrm{i} = 1,2,3, \ldots., \mathrm{n}$

$\mathrm{w} \epsilon \mathfrak{R}^{\mathrm{p}}, \mathrm{u_i} \epsilon \mathfrak{R}^{\mathrm{p}}$are column matrix.p is order of system. At i^{th} iteration u_i is input to system which is tap delay system of order p. e_i is additive white Gaussian noise. But the outlier(impulse noise) is denoted by v_i.so the vectorial representation can be :

$$\mathbf{d} = \mathbf{U}^{\boldsymbol{T}}\boldsymbol{w} + \boldsymbol{e} + \boldsymbol{v} \qquad \textbf{3-11}$$

The problem is to estimate **w** from input **U** and output **d**. The geometrical solution to the problem as below. By finding a point on the span of the input space for which the the distance between the desired point and the point on the input space will be minimum, that is called the projection of the desired to the input space which is denoted by $\mathbf{d_u}$. So the optimum parameter is

$$\mathbf{w} = \boldsymbol{d_u^{-1} U}. \qquad \textbf{3-12}$$

In order to find the point on the span space of the input that will minimize the distance a norm is used. In conventional technique L_2 norm is used but this norm is very sensitive to the impulse noise present in the desired. But Wilcoxon norm and sign Wilcoxon norm are robust to the impulse noise. For estimating the parameter of a system using Wilcoxon norm as cost function gradient based technique has been using equation

$$\widehat{\mathbf{w}}_{\mathbf{i+1}} = \widehat{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\left(\sum_{j=1}^{L} \boldsymbol{\varphi}\left(\boldsymbol{R}\left(\boldsymbol{e}_{iL+j}\right)\right)\mathbf{u}_{iL+j}\right) \qquad \textbf{3-13}$$

In matrix vector multiplication form,

$$\widehat{\mathbf{w}}_{\mathbf{i+1}} = \widehat{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\begin{bmatrix}\mathbf{u}_{iL+1}\\ \mathbf{u}_{iL+2}\\ \vdots\\ \mathbf{u}_{iL+L}\end{bmatrix}\left[\begin{bmatrix}\boldsymbol{\varphi}(\boldsymbol{R}(\boldsymbol{e}_{iL+1}))\\ \boldsymbol{\varphi}(\boldsymbol{R}(\boldsymbol{e}_{iL+2}))\\ \vdots\\ \boldsymbol{\varphi}(\boldsymbol{R}(\boldsymbol{e}_{iL+L}))\end{bmatrix}\right]^{\boldsymbol{T}} \qquad \textbf{3-14}$$

$$\widehat{\mathbf{w}}_{\mathbf{i+1}} = \widehat{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\boldsymbol{U}_i^{\boldsymbol{T}}\boldsymbol{s}_i \qquad \textbf{3-15}$$

Where $\boldsymbol{U}_i^{\boldsymbol{T}} = [\boldsymbol{u}_{iL+1} \quad \boldsymbol{u}_{iL+2} \quad \cdots \quad \boldsymbol{u}_{iL+L}]$

and $\boldsymbol{s}_i = [\boldsymbol{\varphi}(\boldsymbol{R}(\boldsymbol{e}_{iL+1})) \quad \boldsymbol{\varphi}(\boldsymbol{R}(\boldsymbol{e}_{iL+2})) \quad \cdots \quad (\boldsymbol{R}(\boldsymbol{e}_{iL+L}))]$.

In case of sign Wilcoxon norm we can write simillar matrix vector multiplication form like

$$\widehat{\mathbf{w}}_{\mathbf{i+1}} = \widehat{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\boldsymbol{U}_i^{\boldsymbol{T}}\boldsymbol{sign}(\boldsymbol{s}_i) \qquad \textbf{3-16}$$

3.4.2 Sign Regressor Wilcoxon

U_i^T *and* s_i Wilcoxon update equations are acting like input and error in LMS respectively. It is known that sign regressor LMS and sign-sign LMS are faster than the LMS.

$$\widehat{w}_{i+1} = \widehat{w}_i + \mu sign(u_i)e_i$$

Comparing (14) with (12) and taking sign of the U_i^T of (12) we designed the update equation for sign regressor Wilcoxon like below

$$\widehat{\mathbf{w}}_{\mathrm{i+1}} = \widehat{\mathbf{w}}_{\mathrm{i}} + \mu sign(U_i^T)s_i$$

Changing the matrix vector multiplication part of (15) into summation of multiplication form

$$\widehat{\mathbf{w}}_{\mathbf{i+1}} = \widehat{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\left(\sum_{j=1}^{L} \boldsymbol{\varphi}\left(\boldsymbol{R}\left(\boldsymbol{e}_{iL+j}\right)\right)\mathbf{sign}(\mathbf{u}_{iL+j})\right) \qquad \textbf{3-17}$$

3.4.3 Sign Sign Wilcoxon

The update equation for sign-sign LMS is

$$\widetilde{\mathbf{w}}_{\mathbf{i+1}} = \widetilde{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu sign}(\boldsymbol{u_i})\boldsymbol{sign}(\boldsymbol{e_i}) \quad \textbf{3-18}$$

Comparing (17) with (12) and taking sign of the U_i^T of (12) we designed the update equation for sign regressor Wilcoxon like below

$$\widetilde{\mathbf{w}}_{\mathbf{i+1}} = \widetilde{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu sign}(\boldsymbol{U_i^T})\boldsymbol{sign}(\boldsymbol{s_i}) \quad \textbf{3-19}$$

Changing the matrix vector multiplication part of (18) into summation of multiplication form

$$\widetilde{\mathbf{w}}_{\mathbf{i+1}} = \widetilde{\mathbf{w}}_{\mathbf{i}} + \boldsymbol{\mu}\left(\sum_{j=1}^{L} \boldsymbol{sign}\left(\boldsymbol{\varphi}\left(\boldsymbol{R}\left(\boldsymbol{e_{iL+j}}\right)\right)\right)\mathbf{sign}(\mathbf{u}_{\boldsymbol{iL+j}})\right) \quad \textbf{3-20}$$

3.4.4 Simulation Result Discussion

From simulation results it is verified that the convergence speed of proposed techniques Sign-regressor Wilcoxon, sign-sign Wilcoxon are robust against the impulse noise present in desired data and convergence speed are faster than the sign Wilcoxon and Wilcoxon. The performance of the proposed techniques is dependent on the system type used. In this section only simulation results are shown but there is large work to be done like convergence analysis, stability and breakdown point of the algorithms with respect to impulse noise present in the desired data. Since the convergence speed of the proposed techniques are very fast it can apply to the fast varying system.

From simulation results we can conclude that the convergence speed of proposed techniques Sign-regressor Wilcoxon and sign-sign Wilcoxon are robust against the impulse noise present in desired data and convergence speed are faster than the sign Wilcoxon and Wilcoxon. The performances of the proposed techniques are dependent on the system we are using. In this paper only simulation results are shown but there is large work to be done like convergence analysis, stability and breakdown point of the algorithms with respect to impulse noise present in the desired data. It can consider as the future work of the algorithms. Since the convergence speed of the proposed techniques are very fast it can apply to the fast varying system.

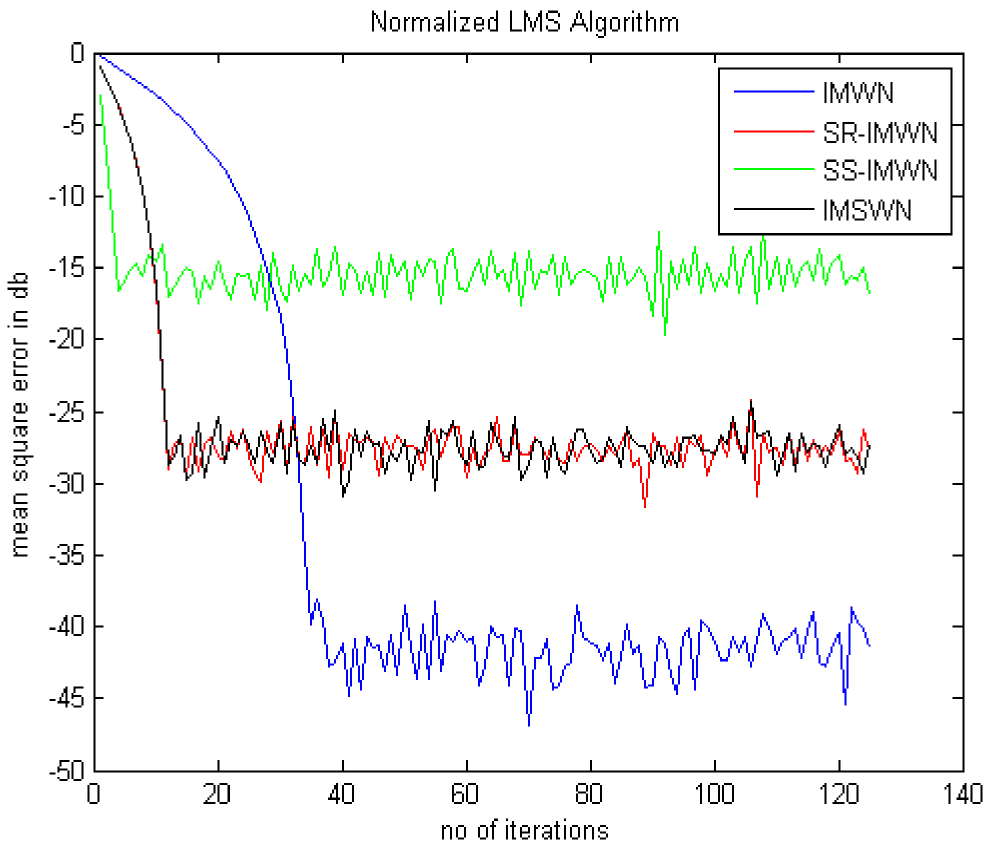
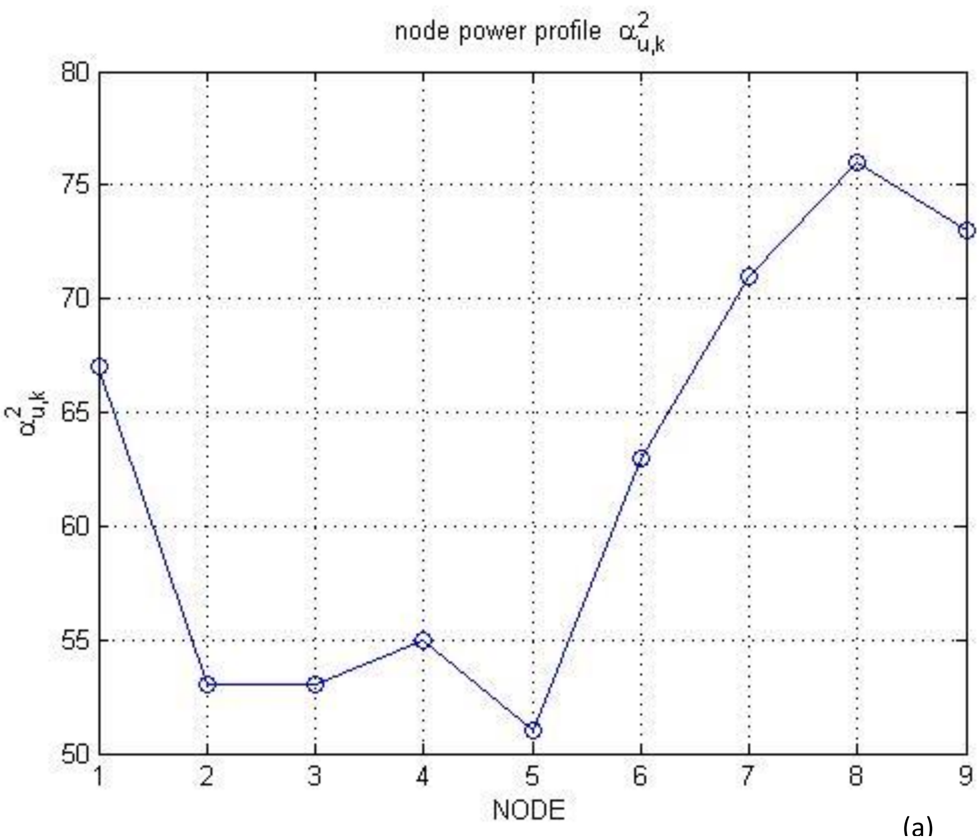
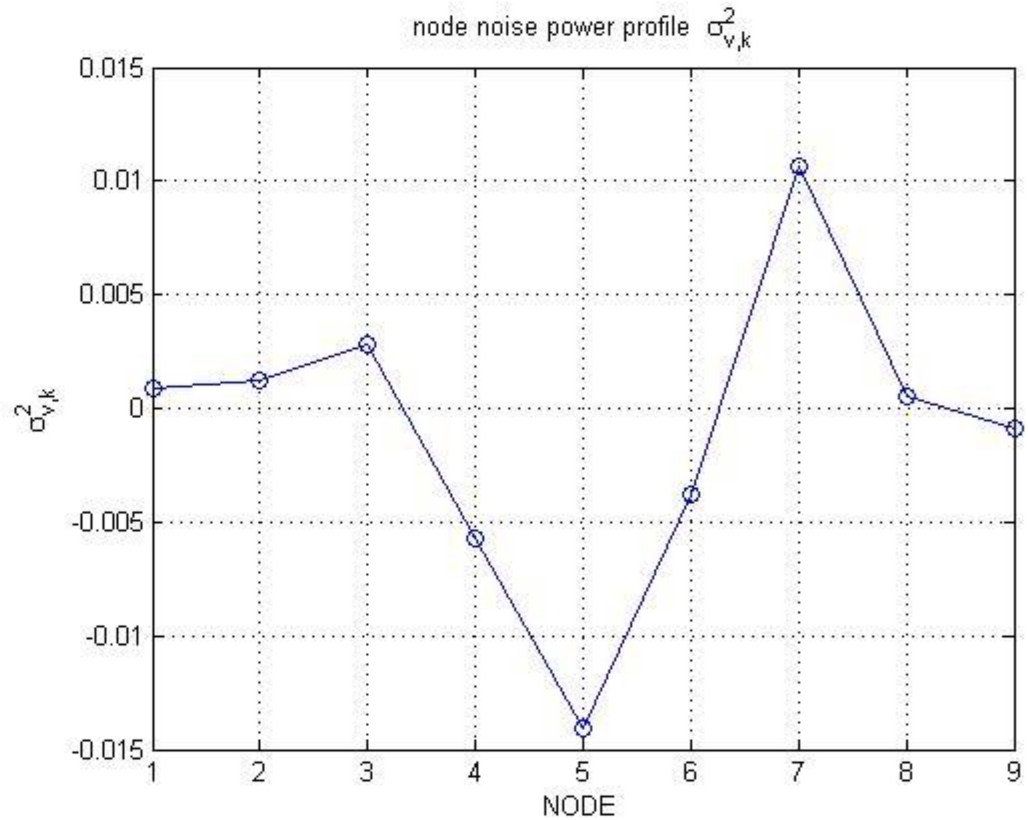


Figure 3-3 wilcoxon norm on incremental strategy

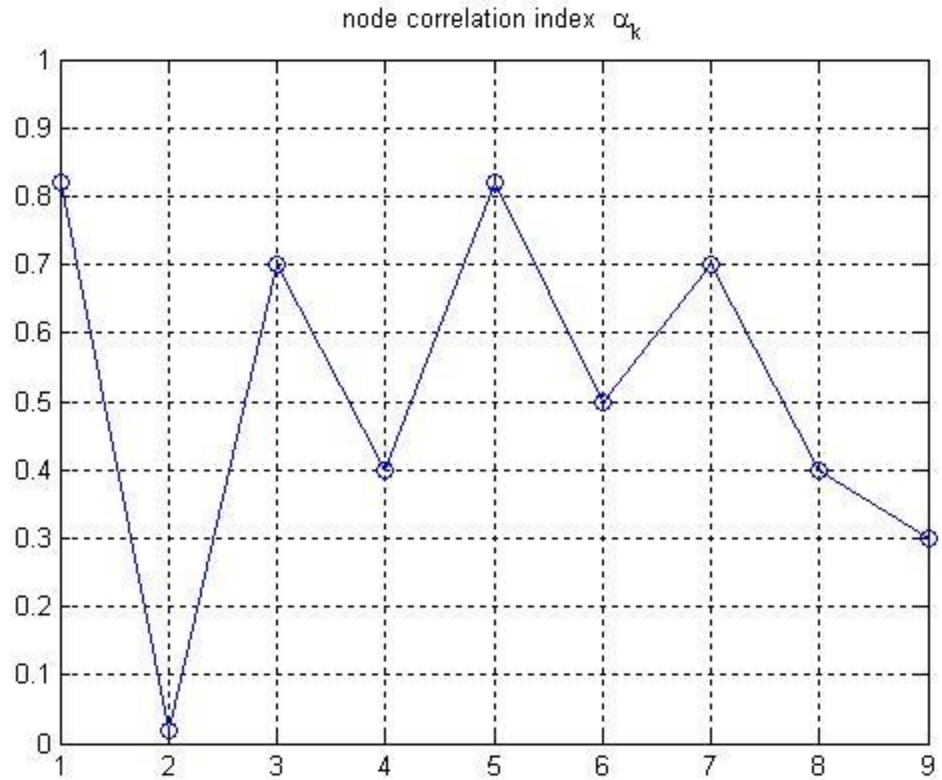
The simulation result is produced for node N=9. The back ground noise power = 10^{-3}. Network statistics are given as under. There are 20 independent experiments are conducted and various error is averaged for display purposes. The error curves are generated from learning process for 1000 iteration. The error are collected over samples and averaged over number of experiments. The global mean square deviation MSD is calculated from averaging $E\|\overline{\Psi}_k^{j-1}\|$ across nodes over 20 experiments. The global excess mean square error EMSE $E\|e_{a,k}(j)\|^2$ where $e_{a,k}(j) = x_{k,j}\overline{\Psi}_k^{j-1}$ mean square error MSE $E\|d_k - x_{k,j}\overline{\Psi}_k^{j-1}\|$ is also conducted as described for MSD. For local error estimation of MSE, EMSE, MSD at node 1 is given in the plot of simulation.


node power profile $\alpha^2_{u,k}$
$\alpha^2_{u,k}$
NODE
80
75
70
65
60
55
50
1
2
3
4
5
6
7
8
9

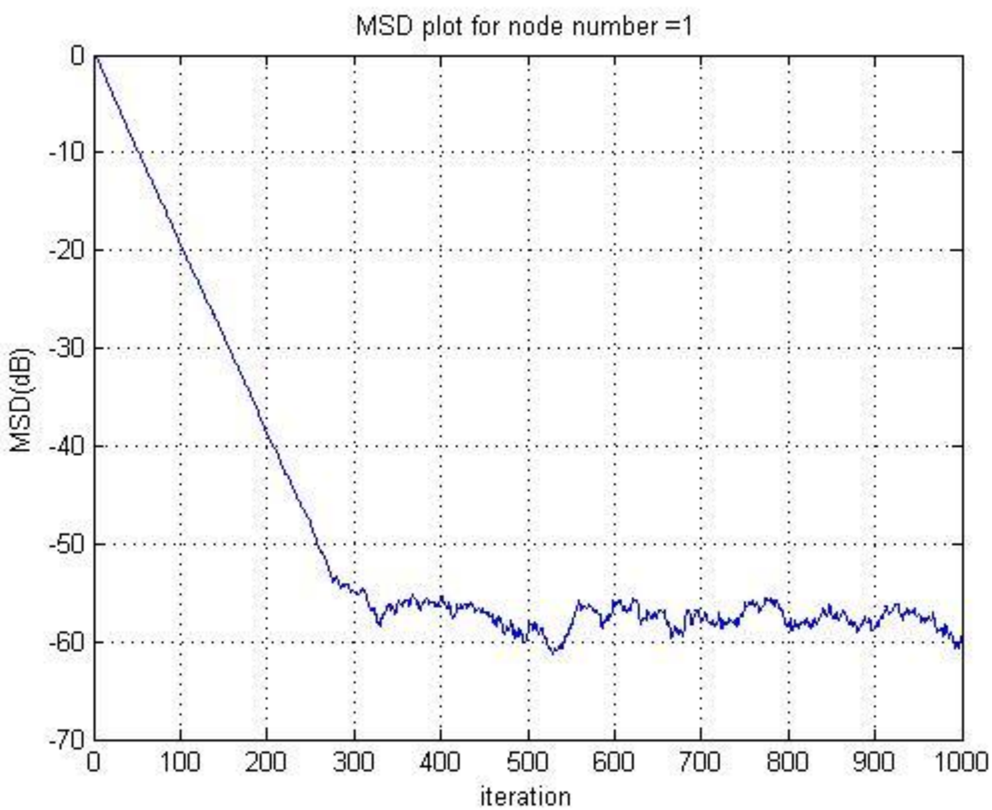
(a)


node noise power profile $\sigma^2_{v,k}$
$\sigma^2_{v,k}$
NODE
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
1
2
3
4
5
6
7
8
9

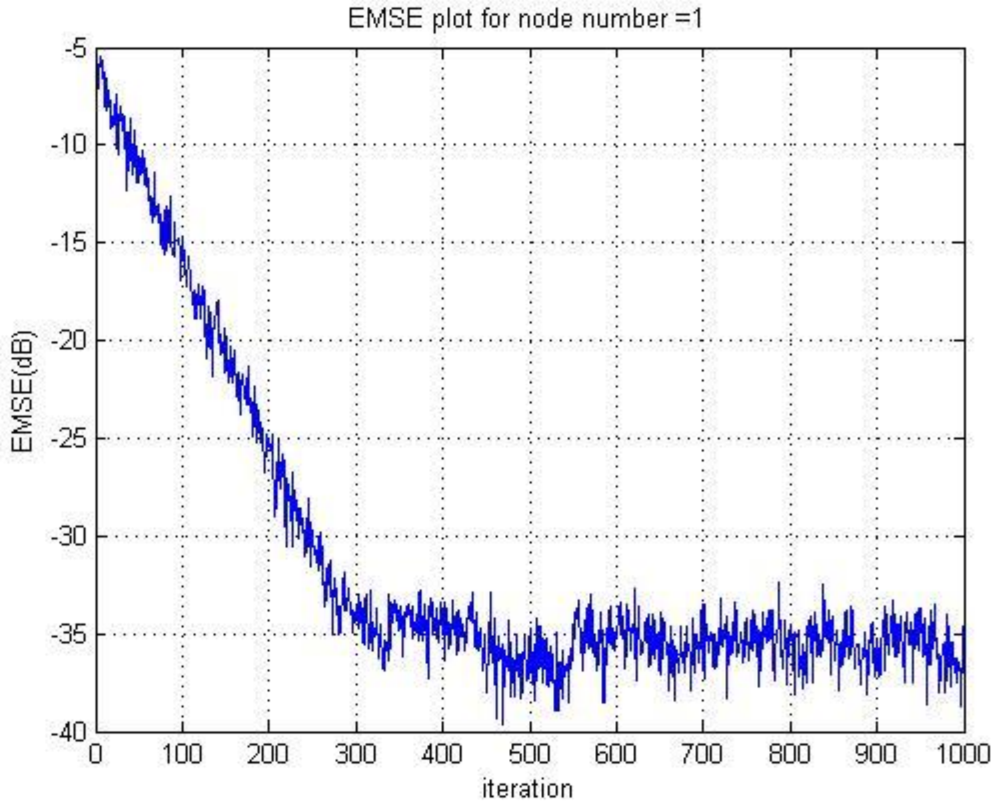
(b)


node correlation index α_k
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
5
6
7
8
9

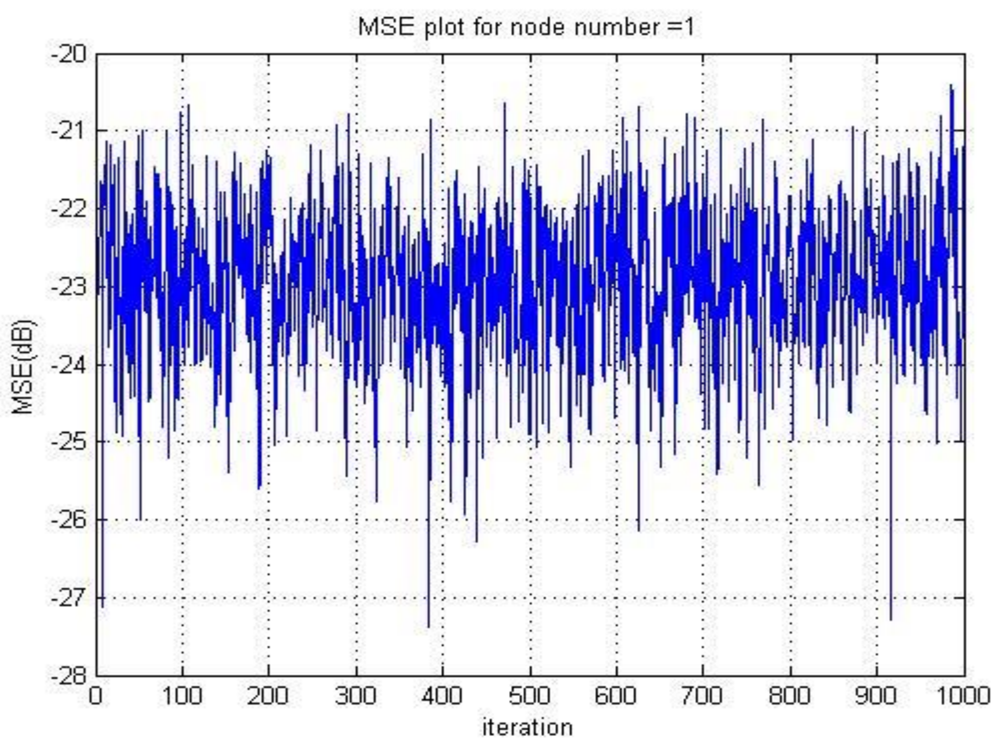
(c)


MSD plot for node number =1
MSD(dB)
0
-10
-20
-30
-40
-50
-60
-70
0
100
200
300
400
500
600
700
800
900
1000
iteration

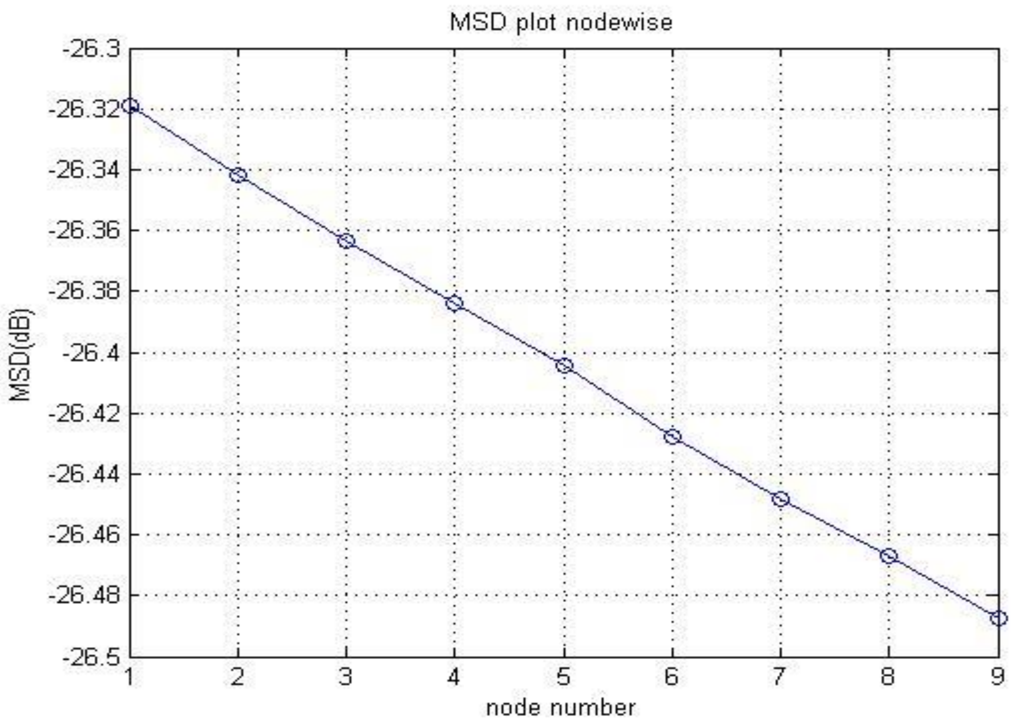
(d)


EMSE plot for node number =1
EMSE(dB)
iteration
-5
-10
-15
-20
-25
-30
-35
-40
0
100
200
300
400
500
600
700
800
900
1000

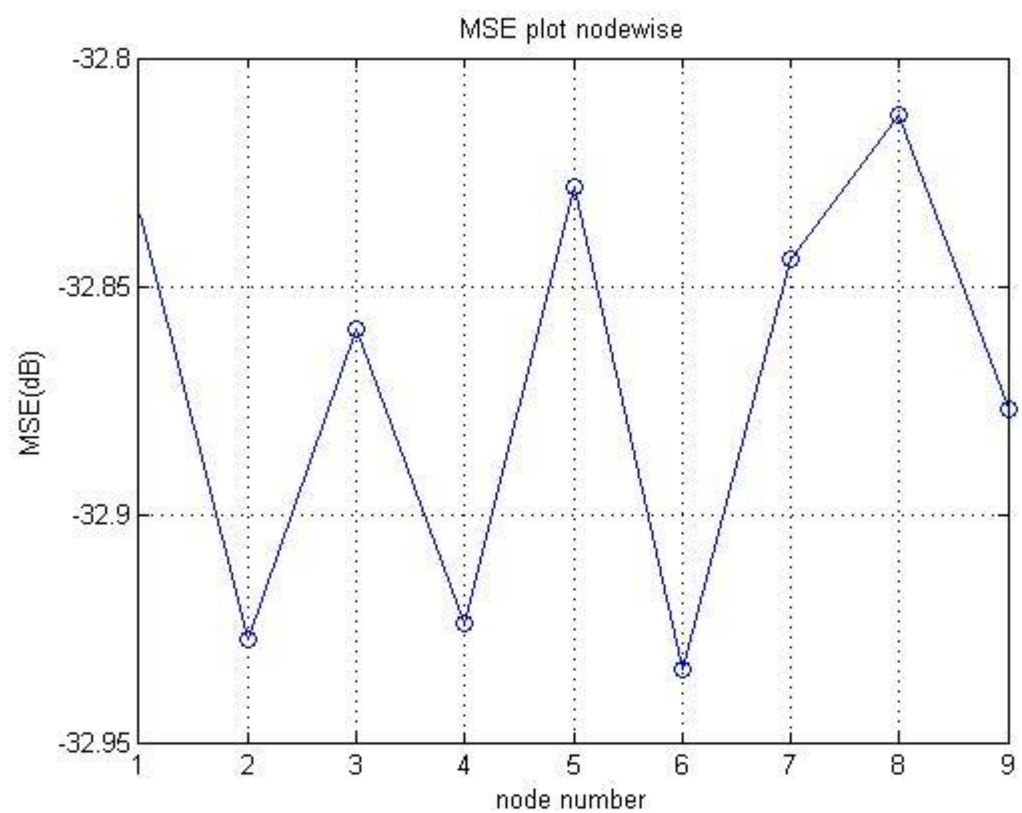
(e)


MSE plot for node number =1
MSE(dB)
iteration
-20
-21
-22
-23
-24
-25
-26
-27
-28
0
100
200
300
400
500
600
700
800
900
1000

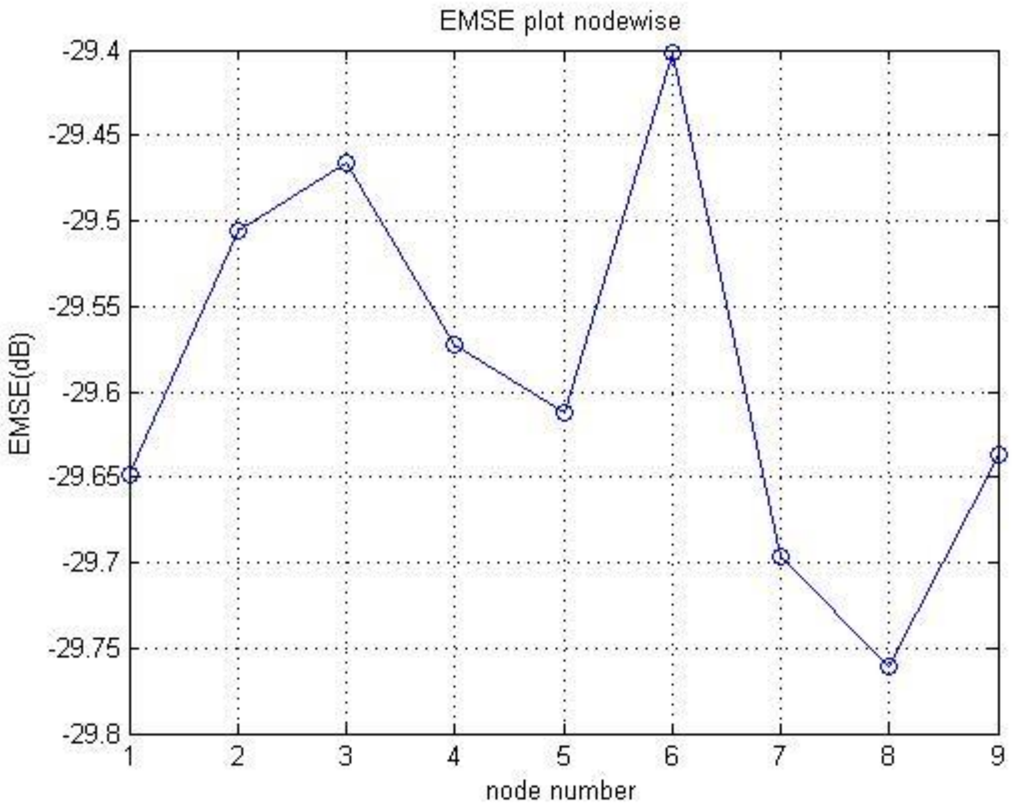
(f)


MSD plot nodewise
MSD(dB)
node number
-26.3
-26.32
-26.34
-26.36
-26.38
-26.4
-26.42
-26.44
-26.46
-26.48
-26.5
1
2
3
4
5
6
7
8
9

(g)


MSE plot nodewise
MSE(dB)
node number
-32.8
-32.85
-32.9
-32.95
1
2
3
4
5
6
7
8
9

(h)



(i)

Figure 3-4 incremental LMS adaptive strategy (a) node power profile (b) noise power profile (c) correlation index (d) MSD at node 1 (e) EMSE at node 1 (f) MSE at node 1 (g) MSD node wise (h) MSE node wise (i) EMSE node wise

3.5 Conclusion

This chapter introduced a distributed incremental approach for estimation in wireless sensor network. The steady-state performance of the incremental LMS algorithm was presented. The spatial energy-conservation arguments were used to study the steady-state performance of the network. The steady-state expression for MSE, EMSE and MSD were derived and was found to be matching very well with simulation results. Contrary to consensus strategy, distributed approximate least-squares solution for a limited number of measurements was proposed, using one measurement per node; the resulting solution is not adaptive.

Chapter 4

Computational Efficient Incremental Minimum Wilcoxon Norm

4. Chapter 4:: Computational Efficient Incremental Minimum Wilcoxon Norm

As an efficient high durable system the power consumption thus computational complexity load should be controlled in adaptive filter implementations. Partial updating of LMS filter coefficients is one of durable methodology in this power maintenance scheme. The problem of distributed estimation, a set of node is required to collectively estimate some parameter of interest from noisy measurement. The problem is useful in several contexts including wireless and sensor networks, where robustness, scalability and low power consumption for longer operating life are desirable features. In the gradual approximation selection of filter coefficient are upgraded in regular fashion [20].

Earlier approach for partial updating is through round robin updating of coefficient subsets (sequential partial updates) and updating all the coefficients at periodic intervals (periodic partial updates). But , these data-independent approaches suffer from convergence rate reduction, which is proportional to the size of coefficient subsets for sequential partial updates and the update frequency for periodic partial updates.so some data dependent approach as stochastic partial updates, M-max updates, selective partial updates, and set membership partial updates are mathematically introduced. The limitation is for the small step size parameter for lower error rate but causing sluggish system response.

4.1 Periodic Partial Update

Basically the periodic updating converges to the optimal solution in a number of iterations by reducing average update complexity of the overall iteration period [21] [22]. Considering an generic adaptive filter:

$$\boldsymbol{w}(\boldsymbol{k}+\mathbf{1})=\boldsymbol{w}(\boldsymbol{k})+\boldsymbol{f}(\boldsymbol{k}), \quad \boldsymbol{k}=\mathbf{0},\mathbf{1},\mathbf{2},\ldots \qquad \textbf{4-1}$$

$$\boldsymbol{adaptive\ filter\ output\ at\ iteration\ k\ is\ y(k)=h\big(w(k),x(k)\big)} \qquad \textbf{4-2}$$

N×1 filter coefficient vector is

$$\boldsymbol{w}(\boldsymbol{k})=[\boldsymbol{w}_{\mathbf{1}}(\boldsymbol{k}),\boldsymbol{w}_{\mathbf{2}}(\boldsymbol{k}),\boldsymbol{w}_{\mathbf{3}}(\boldsymbol{k}),\ldots\ldots\ldots.,\boldsymbol{w}_{\boldsymbol{N}}(\boldsymbol{k})]^{\boldsymbol{T}} \qquad \textbf{4-3}$$

Where

$$\boldsymbol{f}(\boldsymbol{k})=[\boldsymbol{f}_{\mathbf{1}}(\boldsymbol{k}),\boldsymbol{f}_{\mathbf{2}}(\boldsymbol{k}),\boldsymbol{f}_{\mathbf{3}}(\boldsymbol{k}),\ldots\ldots\ldots.,\boldsymbol{f}_{\boldsymbol{N}}(\boldsymbol{k})]^{\boldsymbol{T}} \qquad \boldsymbol{N}\times\mathbf{1}\ \boldsymbol{update\ vector}$$

$$\boldsymbol{x}(\boldsymbol{k})=[\boldsymbol{x}(\boldsymbol{k}),\boldsymbol{x}(\boldsymbol{k}-\mathbf{1}),\ldots\ldots,\boldsymbol{x}(\boldsymbol{k}-\boldsymbol{N}+\mathbf{1})]^{\boldsymbol{T}} \qquad \boldsymbol{N}\times\mathbf{1}\ \boldsymbol{regressor\ vector} \qquad \textbf{4-4}$$

The principle of partial updating is described as

$$\boldsymbol{w}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = \boldsymbol{w}(\boldsymbol{kS}) + \boldsymbol{f}(\boldsymbol{kS}), \quad \boldsymbol{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \dots$$
$$\boldsymbol{w}(\boldsymbol{kS}+\boldsymbol{i}) = \boldsymbol{w}(\boldsymbol{kS}), \quad \boldsymbol{i} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3} \dots \dots .., \boldsymbol{S}-\mathbf{1} \qquad \text{4-5}$$
$$\boldsymbol{y}(\boldsymbol{k}) = \boldsymbol{h}(\boldsymbol{w}(\lfloor \boldsymbol{k}/\boldsymbol{S} \rfloor \boldsymbol{S}, \boldsymbol{x}(\boldsymbol{k}))$$

The application of periodic partial updates to LMS adaption process is

$$\boldsymbol{w}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = \boldsymbol{w}(\boldsymbol{kS}) + \boldsymbol{\mu e}(\boldsymbol{kS})\boldsymbol{x}(\boldsymbol{kS}), \quad \boldsymbol{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \dots \qquad \textbf{4-6}$$

$$\boldsymbol{w}(\boldsymbol{kS}+\boldsymbol{i}) = \boldsymbol{w}(\boldsymbol{kS}), \quad \boldsymbol{i} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3} \dots \dots .., \boldsymbol{S}-\mathbf{1}$$

In N coefficient system M is to be updated at each time index k complexity reduced by factor

$$\boldsymbol{s} = \lceil \boldsymbol{N}/\boldsymbol{M} \rceil \quad \boldsymbol{where} \ \lceil x \rceil \ \boldsymbol{is\ trauncti on\ operator\ on\ x} \qquad \textbf{4-7}$$

From above definition LMS filtering equation is deprived as

$$\boldsymbol{y}(\boldsymbol{k}) = \mathbf{w}^{\boldsymbol{T}}\left(\left\lfloor \frac{\boldsymbol{k}}{\boldsymbol{s}} \right\rfloor \boldsymbol{s}\right)\boldsymbol{x}(\boldsymbol{k}) \qquad \textbf{4-8}$$

From equation (2.6) conforms that the update vector is zero at iterations that are not an integer multiple of S, i.e. k mod S ≠0 where mod is the modulo operator. This is equivalent to having $\boldsymbol{x}$(k) = 0 if k mod S ≠ 0. For sufficiently small step-size parameter, the periodic-partial update LMS can be replaced by the following averaged system:

$$\boldsymbol{w}^{\boldsymbol{a}}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = \boldsymbol{w}^{\boldsymbol{a}}(\boldsymbol{kS}) + \boldsymbol{\mu}\big(\overline{\boldsymbol{p}} - \overline{\boldsymbol{R}}\boldsymbol{w}^{\boldsymbol{a}}(\boldsymbol{kS})\big) \ \boldsymbol{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \dots \dots \qquad \textbf{4-9}$$

For stationary signal domain

$$\overline{\boldsymbol{p}} = \boldsymbol{E}[\boldsymbol{x}(\boldsymbol{kS})\boldsymbol{d}(\boldsymbol{kS})] \qquad \textbf{4-10}$$

$$\overline{\boldsymbol{R}} = \boldsymbol{E}[\boldsymbol{x}(\boldsymbol{kS})\boldsymbol{x}^{\boldsymbol{T}}(\boldsymbol{kS})] \qquad \textbf{4-11}$$

are the cross-correlation vector between the periodic-partial-update regressor vector and the corresponding desired filter response, and the autocorrelation matrix of the periodic-partial-update regressor, respectively. The autocorrelation matrix of the periodic-partial-update regressor $\bar{R}$ is identical to the autocorrelation matrix of the input signal $R = E[x(k)x^T(k)]$. However, because the adaptive filter coefficients are updated every Sth iteration, the periodic-partial-update LMS will take S times as long as the full-update LMS to converge. The averaged system with periodic partial updates is a steepest descent algorithm which produces the optimum (minimum mean-square error (MSE)) solution:

$$\boldsymbol{w_0} = \overline{\boldsymbol{R}}^{-\mathbf{1}}\overline{\boldsymbol{p}} \qquad \textbf{4-12}$$

Relationship between $\mathbf{w_0}$ and w^{a}(kS) coefficient error

$$\Delta\boldsymbol{w}^{\boldsymbol{a}}(\boldsymbol{kS}) = \boldsymbol{w_0} - \boldsymbol{w}^{\boldsymbol{a}}(\boldsymbol{kS}) \qquad \textbf{4-13}$$

$$\Delta w^a\big((k+1)S\big) = \Delta w^a(kS) - \mu\big(\bar{p} - \bar{R}w^a(kS)\big)$$

$$= (\mathbf{I} - \boldsymbol{\mu}\bar{\boldsymbol{R}})^{\boldsymbol{k+1}}\Delta \boldsymbol{w^a}(\mathbf{0}) \qquad \textbf{4-14}$$

Applying similarity transformation to $\bar{R}$ to diagonalize it results in:

$$\boldsymbol{Q^T\bar{R}Q} = \underbrace{\begin{bmatrix} \boldsymbol{\lambda_1} & \dots & \mathbf{0} \\ \vdots & \boldsymbol{\lambda_2} & \vdots \\ \mathbf{0} & \dots & \boldsymbol{\lambda_N} \end{bmatrix}}_{\boldsymbol{\Lambda}} \qquad \textbf{4-15}$$

Q is unitary matrix and λ_i are eigenvalues of regressor $\bar{R}$.

$$\boldsymbol{Q^T}\Delta \boldsymbol{w^a}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = (\mathbf{I} - \boldsymbol{\mu Q^T\bar{R}Q})\boldsymbol{Q^T}\Delta \boldsymbol{w^a}(\boldsymbol{kS}) \qquad \textbf{4-16}$$

$$\boldsymbol{v^a}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = (\mathbf{I} - \boldsymbol{\mu\Lambda})\boldsymbol{v^a}(\boldsymbol{kS}) \qquad \textbf{4-17}$$

$$\boldsymbol{v^a}\big((\boldsymbol{k}+\mathbf{1})\boldsymbol{S}\big) = (\boldsymbol{I} - \boldsymbol{\mu\Lambda})\boldsymbol{v^a}(\mathbf{0})$$

$$= \begin{bmatrix} (\mathbf{1}-\boldsymbol{\mu\lambda_1})^{\boldsymbol{k+1}} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & (\mathbf{1}-\boldsymbol{\mu\lambda_2})^{\boldsymbol{k+1}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{0} & \dots & \dots & (\mathbf{1}-\boldsymbol{\mu\lambda_N})^{\boldsymbol{k+1}} \end{bmatrix} \boldsymbol{v^a}(\mathbf{0}) \qquad \textbf{4-18}$$

This expression describes the evolution of the coefficient errors rotated by $\boldsymbol{Q}^T$ for the averaged system. The convergence properties of the full-update and partial-update LMS algorithms are determined from the respective correlation matrices of the coefficient update vector.

4.2 Sequential Partial Update

In contrast to periodic update where complete updating of vector is delayed due to entire vector completion, a set of parameters is modulated on each iteration. The coefficient subset is determined round ribbon fashion [21]. Therefore a generic adaptive filter yields:

$$\boldsymbol{y}(\boldsymbol{k}) = \boldsymbol{h}\big(\boldsymbol{w}(\boldsymbol{k}), \boldsymbol{x}(\boldsymbol{k})\big)$$

$$\boldsymbol{w}(\boldsymbol{k}+\mathbf{1}) = \boldsymbol{w}(\boldsymbol{k}) + \boldsymbol{I_M}(\boldsymbol{k})\boldsymbol{f}(\boldsymbol{k}), \quad \boldsymbol{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, \dots$$

$$\boldsymbol{I_M}(\boldsymbol{k}) = \begin{bmatrix} \boldsymbol{i_1}(\boldsymbol{k}) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{i_2}(\boldsymbol{k}) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \ddots & \boldsymbol{i_N}(\boldsymbol{k}) \end{bmatrix}, \qquad \textstyle\sum_{j=1}^{N} \boldsymbol{i_j}(\boldsymbol{k}) = \boldsymbol{M}, \qquad \boldsymbol{i_j}(\boldsymbol{k})\boldsymbol{\epsilon}[\mathbf{0}, \mathbf{1}] \qquad \textbf{4-19}$$

$I_M(k)$ is coefficient selection matrix. Sequential partial update is implemented by separating B(=N/M) M-subsets of S={1,2,….,N} as 2 conditions is satisfied.

- The union of B M-subsets are S so that no adaptive filter coefficient is left out.

- No M-subset pairs share a common member, which ensures that each coefficient gets an equal chance of update.

In terms of M-subsets $\mathcal{J}_{1,}\mathcal{J}_2 \ldots \ldots \mathcal{J}_{\boldsymbol{B}}$satisfying the above conditions, the coefficient selection matrix for sequential partial updates can be expressed as

$$\boldsymbol{I_M(k)} = \begin{bmatrix} \boldsymbol{i_1(k)} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{i_2(k)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \ldots & \ddots & \boldsymbol{i_N(k)} \end{bmatrix}, \quad \boldsymbol{i_j(k)} = \begin{cases} \boldsymbol{1} & \boldsymbol{if\ j \in \mathcal{J}_{k\ mod\ B+1}} \\ \boldsymbol{0} & \boldsymbol{otherwise} \end{cases} \qquad \textbf{4-20}$$

Updating M coefficients in an adaptive filter of length N at every iteration leads to a complexity reduction in the adaptation process roughly proportional to B. The resource constraints placed on the number of coefficients that can be updated is therefore easily accommodated by sequential partial updates. In practical implementations the choice of M-subsets is guided by software considerations. The usual choice is to partition the adaptive filter coefficient vector into B vectors of length M:

$$\mathbf{w(k)} = [\boldsymbol{w_1(k)}\ \boldsymbol{w_2(k)}\ \boldsymbol{w_3(k)} \ldots \ldots \boldsymbol{w_B(k)}]^{\boldsymbol{T}} \qquad \textbf{4-21}$$

with corresponding update partitions

$$\mathbf{f(k)} = [\boldsymbol{f_1(k)}\ \boldsymbol{f_2(k)}\ \boldsymbol{f_3(k)} \ldots \ldots \boldsymbol{f_B(k)}]^{\boldsymbol{T}} \qquad \textbf{4-22}$$

In a round-robin fashion as illustrated in Figure 2.4. This corresponds to having

$$\begin{aligned} \mathcal{J}_1 &= \{\boldsymbol{1, 2 \ldots . M}\} \\ \mathcal{J}_2 &= \{\boldsymbol{M+1, M+2, \ldots ., 2M}\} \qquad \textbf{4-23} \\ &\vdots \\ \mathcal{J}_B &= \{\boldsymbol{(B-1)M+1, (B-1)M+2, \ldots ., N}\} \end{aligned}$$

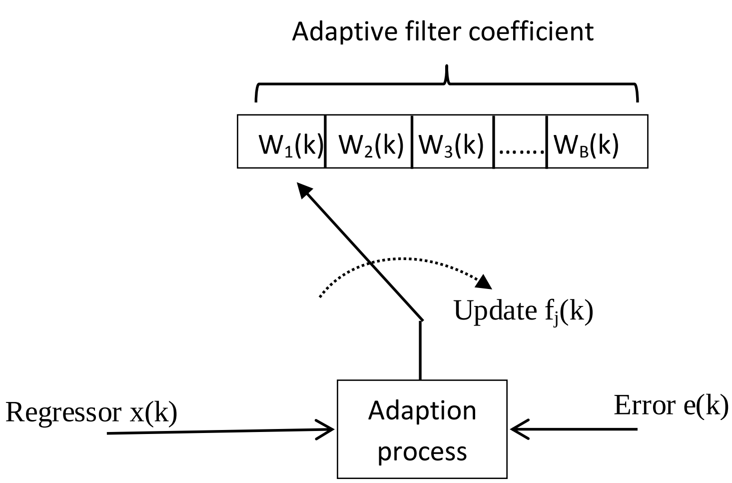


Figure 4-1 Sequential partial updates using partitions of the adaptive filter coefficient vector

The sequential partial update LMS algorithm is given by:

$$\boldsymbol{w(k+1) = w(k) + e(k) I_M(k) x(k) \quad k = 0, 1, 2, 3, \dots} \qquad \textbf{4-24}$$

$$\boldsymbol{y(k) = w^T(k) x(k)} \qquad \textbf{4-25}$$

For a sufficiently small step-size μ, the equivalent averaged system is:

$$\boldsymbol{w^a(k+1) = w^a(k) + \mu\big(\mathrm{p_M} - R_M w^a(k)\big) \quad k = 0, 1, 2, \dots\dots} \qquad \textbf{4-26}$$

Where $\boldsymbol{p_M = E[I_M(k) x(k) d(k)]}$

$$\boldsymbol{R_M = E[I_M(k) x(k) x^T(k)]} \qquad \textbf{4-27}$$

If the input signal is stationary, it is easy to show that the correlation matrix $\mathbf{R_M}$ can be expressed in terms of the autocorrelation matrix of the input signal R=E{x(k) x^T(k)} as $\mathbf{R_M}$=R/B.

This implies that, similar to periodic partial updates, the method of sequential partial updates suffers from reduced convergence rate proportional to B when it is applied to the LMS algorithm.

4.3 Stochastic Partial Updates

In stochastic partial update [23], which is a randomized version of sequential partial updates, the adaptive filter coefficient subsets are selected randomly rather than in a deterministic fashion. The main advantage of stochastic partial updates is to eliminate instability problems experienced by sequential partial updates for certain non-stationary inputs. The method uses following coefficient matrix:

$$\boldsymbol{I_M(k)} = \begin{bmatrix} \boldsymbol{i_1(k)} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{i_2(k)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \dots & \ddots & \boldsymbol{i_N(k)} \end{bmatrix} \quad \boldsymbol{i_j(k)} = \begin{cases} \mathbf{1}, & \boldsymbol{if\ j \epsilon \mathfrak{I}_{m(k)}} \\ \mathbf{0}, & \boldsymbol{otherwise} \end{cases} \qquad \textbf{4-28}$$

Where m (k) is an independent random process with probability mass function:

$$\boldsymbol{Pr\{m(k) = i\} = \pi_i, \quad i = 1, 2, \dots\dots, B, \quad \textstyle\sum_{i=1}^{B} \pi_i = 1} \qquad \textbf{4-29}$$

As coefficient subset are portioned so there is no loss of generality. For a stationary input signal x(k), the coefficient update correlation matrix for the stochastic-partial-update LMS algorithm can be expressed as:

$$\boldsymbol{R_M = E[I_M(k) x(k) x^T(k)]}$$

$$= E\left\{\begin{bmatrix}\pi_1 x_1(k)\\ \pi_2 x_2(k)\\ \vdots \\ \pi_B x_B(k)\end{bmatrix} x^T(k)\right\} \qquad \textbf{4-30}$$

Where $x_i(k), i=1,2,3,\ldots B$ are partition of regressor vector corresponding to subsets $\mathcal{I}_i$.

$$x(k) = \begin{bmatrix} x(k) \\ \vdots \\ x(k-M+1) \\ \ldots\ldots\ldots\ldots\ldots.. \\ x(k-M) \\ \vdots \\ x(k-2M+1) \\ \ldots\ldots\ldots\ldots\ldots.. \\ \vdots \\ \ldots\ldots\ldots\ldots\ldots.. \\ x(k-(B-1)M+1) \\ \vdots \\ x(k-N+1) \end{bmatrix} = \begin{bmatrix} x_1(k) \\ \ldots\ldots\ldots \\ x_2(k) \\ \ldots\ldots\ldots \\ \vdots \\ \ldots\ldots\ldots \\ x_B(k) \end{bmatrix} \qquad \textbf{4-31}$$

For non-stationary inputs the coefficient update correlation matrix takes the form:

$$R_M = \lim_{k\to\infty} \frac{1}{k} \sum_{k=0}^{k-1} I_M(k) x(k) x^T(k) \qquad \textbf{4-32}$$

$$R_M = \lim_{k\to\infty} \frac{1}{k} \begin{bmatrix}\pi_1 x_1(k)\\ \pi_2 x_2(k)\\ \vdots \\ \pi_B x_B(k)\end{bmatrix} x^T(k) \qquad \textbf{4-33}$$

Non-uniform probability masses π_i are often not desirable since they tend to increase the eigenvalue spread of $\mathbf{R_M}$. The preferred choice is, therefore, a uniform probability mass function giving $\pi_i=1/B, i=1,2,\ldots.,B$. In this case the coefficient update correlation matrix becomes:

$$R_M = \frac{R}{B} \qquad \textbf{4-34}$$

For both stationary and non-stationary inputs condition. As we have seen previously, periodic and sequential-partial-update algorithms do not have this property since their coefficient update correlation matrix is not always related to **R** by (4.63) for non-stationary inputs. Based on (4.30) one can conclude that if the full-update LMS algorithm is stable for a given input signal, so is the stochastic-partial-update LMS. This feature of stochastic partial updates is highly desirable mainly because of its stability implications. In terms of convergence rate, the stochastic-partial-update LMS is still slower than the full-update LMS by a factor of B as a consequence of scaling down of the autocorrelation matrix **R**. The complexity reduction achieved by stochastic partial updates is the same as sequential partial updates if one ignores the overheads for generation of the random signal m(k).

4.4 M-MAX Updates

One of the efficient data dependent partial update is by finding M largest magnitude update vector entries. The method of M-max updates yields:

$$\boldsymbol{y(k) = h\big(w(k), x(k)\big)} \qquad \textbf{4-35}$$

$$\boldsymbol{w(k+1) = w(k) + I_M(k) f(k), \quad k = 0, 1, 2, \ldots}$$

Where the coefficient selection matrix $\mathbf{I_M}$(k) as:

$$\boldsymbol{I_M(k)} = \begin{bmatrix} \boldsymbol{i_1(k)} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{i_2(k)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \ldots & \ddots & \boldsymbol{i_N(k)} \end{bmatrix},$$

$$\boldsymbol{i_j(k)} = \begin{cases} \mathbf{1}, & \boldsymbol{if\ \left|f_j(k)\right|\ j \epsilon\ \mathbf{max}_{1 \le l \le N}(|f_l(k)|, M)} \\ \mathbf{0}, & \boldsymbol{otherwise} \end{cases} \qquad \textbf{4-36}$$

The method of M-max updates [24] is similar to sequential partial updates in that both approaches ‘decimate’ the update vector. However, the main difference between the two approaches lies in the way coefficient subsets are selected. In M-max updates the coefficient selection criterion requires the magnitude of update vector entries to be ranked. As different from deterministic round-robin selection, in each iteration the M-subset of adaptive filter coefficients corresponding to the M largest magnitude update vector entries are updated. This coefficient selection scheme finds the subset of update vector entries which is deemed to make the most contribution to the convergence of the adaptive filter. The number of coefficients M is chosen to remain within the bounds of affordable complexity.

In M-max updates the coefficient selection criterion requires the magnitude of update vector entries to be ranked. As different from deterministic round-robin selection, in each iteration the M subset of adaptive filter coefficients corresponding to the M largest magnitude update vector entries are updated. This coefficient selection scheme finds the subset of update vector entries which is deemed to make the most contribution to the convergence of the adaptive filter. The number of coefficients M is chosen to remain within the bounds of affordable complexity.

4.5 Simulation result

The simulation experiment is operated for individual node N=9. The back ground noise power = 10^{-3} keeping SNR ration to be -30 dB around node. Simulated network experimental statistics are given as under. There are 15 independent experiments are conducted and mean square error is averaged for graphical display. The error curves are generated from learning process for 1800 iteration. The error are collected over samples and averaged over number of experiments. The mean square error MSE $E\left\|d_k - x_{k,j}\overline{\psi}_k^{j-1}\right\|$ is also conducted for node 1. The tap size of the system is M=10 from which N=4 is taken at a time in incremental update method so for partial updating technique of full, partial, sequential, stochastic, M max updating technique is studied by system plot. The convergence time is increased the factor of truncation M/N.

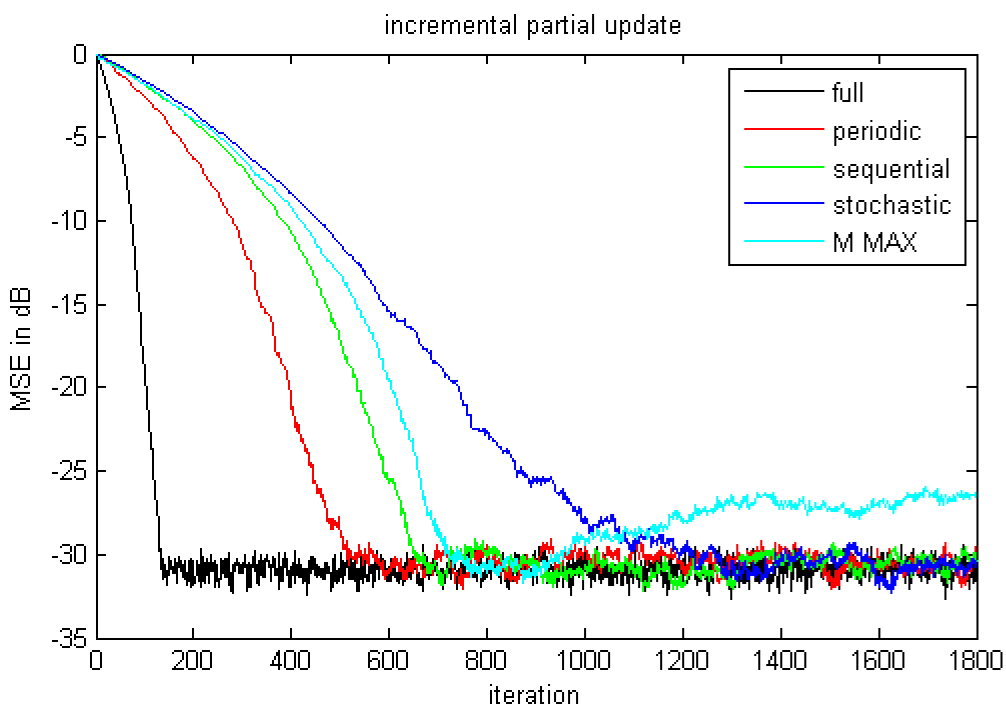


Figure 4-2 partial update where M=10 ; N=4

4.6 Power Consumption Calculation

Considering a system having M tap length provided N is to be updated at each iteration. The complexity arises as number of arithmetic calculation increases resulting high power consumption. In this section overall parameter calculation is provided for simpler understanding.

<u>Error estimation</u>

In every calculation comprising the equation

$$e_{i+1} = d_{i+1} - U_{i+1}^{T} w_i$$

For $U_{i+1}^{T} w_i$ there are M number of multiplication and M−1 number of additions. This result is summarized with d_{i+1} providing an extra addition operator. So for every error e_{i+1} there are M multiplication and (M−1) +1=M addition is required.

Steepest descent wiener hopf solution

As LMS algorithm is $w_{i+1} = w_i + \mu I_N x_{i+1} e_i$

In regressor vector N tap has to be updated so (N+1)+1 Multiplication and N addition.

Wilcoxon norm for block length ‘l’

For score function the error regressor $[e_1\ e_2\ e_3 \ldots e_l]$ is

$\|e\|_{w_N} = \sum_l \sqrt{12}\left(\frac{R(e_i)}{l+1} - 0.5\right) e_i$. Where $R(e_i) = \sum_{k=1}^{L} I\left(e_i - e_j\right)$ as rank $I_x = \begin{cases} 1, x \geq 0 \\ 0, \text{otherwise} \end{cases}$

Therefore R(ei) estimating error sequence by 0.5*L*(L-1) each error regressor lp number of multiplication and lp number of addition.

For score function at each error L division L subtraction is applied.

From the above discussion in 1st iteration L^2 numbers of addition

LP numbers of multiplication

L numbers of divisions

In other iteration L*(L-1) number of subtraction

LP number of multiplication

LP number of addition is required.

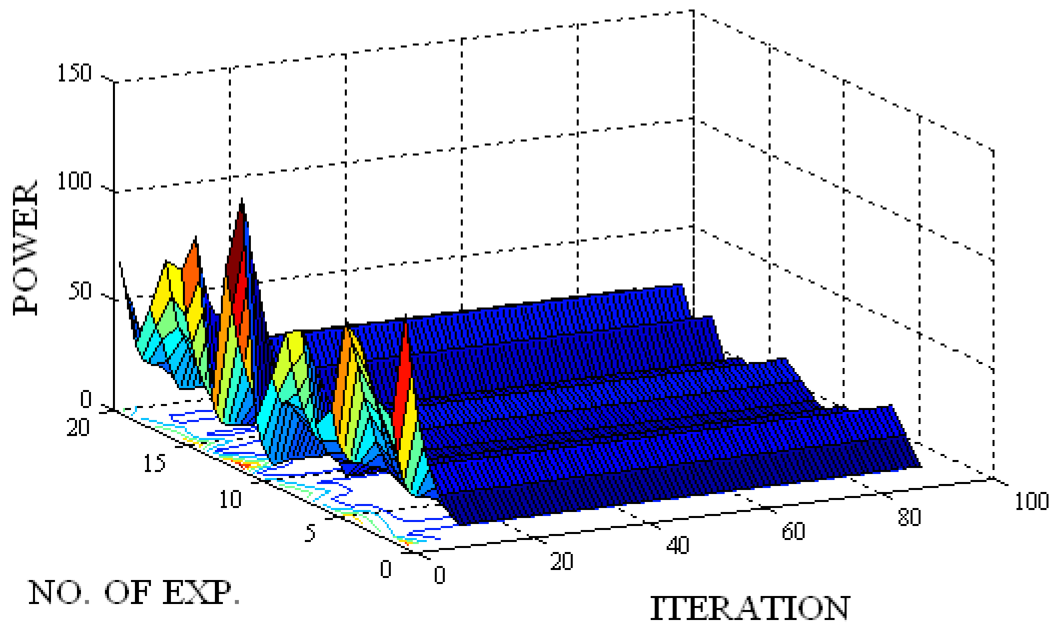


Figure 4-3 Power profile of node on partial updating

Chapter 5

Computational efficient Diffusion Minimum Wilcoxon Norm

5. CHAPTER 5::Computational Efficient Diffusion Minimum Wilcoxon Norm

5.1 Introduction

A wireless sensor network consists of groups of sensors or nodes using wireless links to perform distributed sensing tasks by coordinating among themselves [25]. Distributed signal processing deals with the extraction of information from data collected at nodes, that are distributed over a geographical area [26] [27] [28]. Each node in a network collects noisy observations related to parameter. The nodes would then interact with their neighbors in a certain manner according to the network topology either incremental way or by diffusion. In a traditional centralized solution, the node in the network collects observations and conveys then to central processor where they would be fused and the vector of parameters estimated, then broadcast the result back to the individual node. This mode of operation requires a powerful central processor and more communication between nodes and the processor. In addition, a centralized solution may limit the ability of the nodes to adapt in real time.

In this chapter, a new type of cooperative algorithms is considered [10] that adopt diffusion protocol, in which nodes from the same neighborhood are allowed to communicate with each other. A network is more efficient if it requires less communication between nodes to estimate some vector of parameters.

5.2 Diffusion LMS methodology

To estimate an M X 1 unknown vector w^0 from measurements collected at N nodes spread over a network. Each node k has access to time realizations $\{d_k(j),\ u_{k,j}\}$of zero mean random [7]data$\{d_k, u_k\}$,k = 1, 2, 3, ...N, where $d_k(j)$ is scalar measurement and $u_{k,j}$ a 1 × M regression row vector, both at time j is given as,

$$\mathbf{u_{k,j}} = [\mathbf{u_k(j)}, \mathbf{u_k(j-1)}, \dots \dots, \mathbf{u_k(j-M+1)}]$$

The regression and measurement data across all nodes into two global matrices where we drop the time index for compactness of notation.

$$\mathbf{U_e} = [\mathbf{u_1}, \mathbf{u_{2,}} \dots \dots, \mathbf{u_N}] \qquad \textbf{5-1}$$

$$\mathbf{d} = [\mathbf{d_1}, \mathbf{d_{2,}} \dots \dots, \mathbf{d_N}]$$

The autocorrelation matrix $R_u = E|U_c^* U_c|$ cross-correlation matrix $R_{du} = E|U_c^* d|$.Where E is the expectation operator. For optimizing the solution

$$\min_{\mathbf{w}} \mathbf{E}\|\mathbf{d} - \mathbf{U_c w}\|^2 \qquad \textbf{5-2}$$

The optimal solution for w^0 of (4.1) satisfies the orthogonal condition

$$\mathbf{E U_c^* (d - U_c w^0) = 0} \qquad \textbf{5-3}$$

$$\mathbf{w^0 = R_u^{-1} R_{du}} \qquad \textbf{5-4}$$

For reference introducing block diagram matrix

$$\mathbf{U=diag\{U_1, U_2, U_3,\ldots\ldots, U_N\}} \qquad \mathbf{(N \times NM)} \qquad \textbf{5-5}$$

$$\mathbf{Q=[I_M, I_M, I_M,\ldots\ldots, I_M]^T} \qquad \mathbf{(NM \times M)} \qquad \textbf{5-6}$$

Where I_M is $M \times M$ identity matrix which is related Uc as

$$\mathbf{U_c=UQ.} \qquad \textbf{5-7}$$

5.3 Diffusion LMS Algorithm

The objective is to develop an adaptive distributed algorithm that allows cooperation among the nodes through limited local communications and gives the approximate solution w^0 of (4.4). In addition it should deliver a good estimate of that vector at every node in the network. Here we develop a diffusion protocol where every node k in the network continuously combines estimates from its neighborhood. Specifically, at any given time j−1, we assume that node k has access to a set of unbiased estimates$\{\psi_k^{j-1}\}_{k\epsilon\mathcal{N}_k}$ from its neighborhood $\mathcal{N}_k$.Which is defined as the set of all nodes connected to it, including itself. The estimates $\{\psi_k^{j-1}\}_{k\epsilon\mathcal{N}_k}$ are generally noisy versions of w^0

$$\boldsymbol{\{\psi_k^{j-1}\}_{k\epsilon\mathcal{N}_k} = w^0 - \bar{\psi}_k^{j-1}} \qquad \textbf{5-8}$$

for some error ψ_k^{j-1}. These local estimates are fused at node k, yielding

$$\boldsymbol{\phi_k^{j-1} = f_k(\psi_l^{j-1}; l\epsilon\mathcal{N}_{k,j-1})} \qquad \textbf{5-9}$$

for some local combiner function f_k. Here we employ a linear combiner, and replace f_k by some weighted combination as

$$\boldsymbol{\phi_k^{j-1} = \sum_{l\epsilon\mathcal{N}_{k,j-1}} c_{kl}\psi_l^{j-1}} \quad , \quad \boldsymbol{\phi_k^{-1} = 0} \qquad \textbf{5-10}$$

for some combination coefficients $c_{kl} \geq 0$ to be determined from the network topology.

$$\sum_{l} \mathbf{c_{kl}} = \mathbf{1}, \quad \mathbf{l \epsilon \mathcal{N}_{k,j-1}} \ \ \mathbf{for} \ \forall \ \mathbf{k} \qquad \mathbf{5\text{-}11}$$

Once an aggregate estimate ϕ_k^{j-1} for w^0, and in order to foster adaptively system analysis, we subsequently fuse the resulting estimateϕ_k^{j-1}into the local adaptive process, so that it can rapidly respond to changes in its neighborhood and update it toψ_k^j. Analysis and simulation will show that this scheme leads to a robust distributed adaptive system that achieves smaller error levels in steady-state than its non-cooperative counterpart.

$$\boldsymbol{\psi_k^j} = \boldsymbol{\phi_k^{j-1}} + \mu_{\mathbf{k}} \mathbf{u_{k,j}^*}(\mathbf{d_k}(\mathbf{j}) - \mathbf{u_{k,j}} \boldsymbol{\phi_k^{j-1}}) \qquad \mathbf{5\text{-}12}$$

For local step sizes μ_k, the combiners may be nonlinear or even time-variant, to reflect the changing topologies. This can be implemented by assuming the neighborhood $\mathcal{N}_k$to be time variant. The resulting adaptive network is a peer-to-peer estimation framework and robust to node and link failures.

As per linear combiner formulation by LMS algorithm

$$\boldsymbol{\phi_k^{j-1}} = \sum_{\mathcal{N}_{k,j-1}} \boldsymbol{c_{kl}} \, \boldsymbol{\psi_l^{j-1}} \ \ \boldsymbol{where} \ \boldsymbol{\phi_k^{-1}} = \mathbf{0} \qquad \mathbf{5\text{-}13}$$

$$\boldsymbol{\psi_l^i} = \boldsymbol{\phi_k^{j-1}} + \mu_{\mathbf{k}} \mathbf{u_{k,j}^*}(\mathbf{d_k}(\mathbf{j}) - \mathbf{u_{k,j}} \boldsymbol{\phi_k^{j-1}}) \qquad \mathbf{5\text{-}14}$$

5.4 Diffusion LMS algorithm formulation

The approximation of diffusion equation 5.13 can be realized by two different approaches. One in which adaptation process is achieved before combination of neighbor parameter while the other one performs LMS combination prior to adaption. As their operation concern these are named as ATC diffusion strategy and CTA diffusion strategy. The details procedure of each algorithm is described here below.

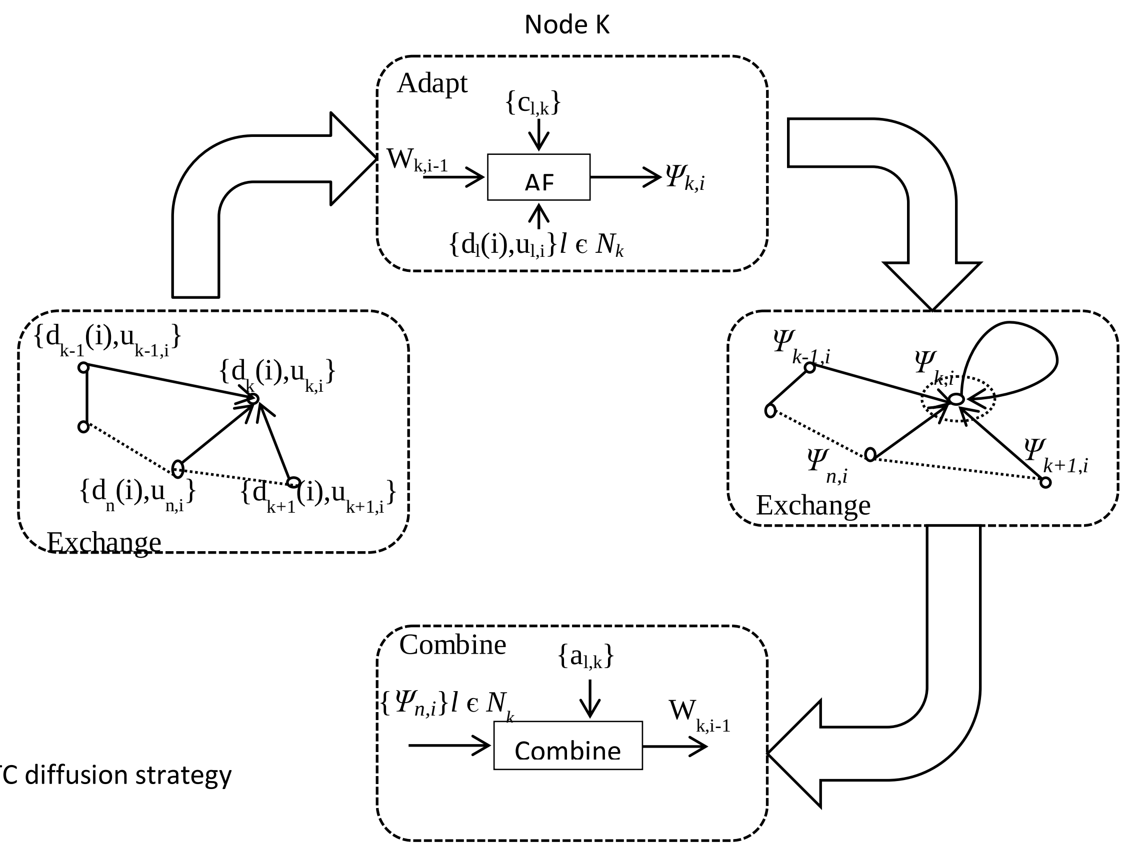
Star with $\{w_{l,-1} = 0\}$ for all l .for real non-negative real coefficients$\{c_{l,k}, a_{l,k}\}$ for each time i≥0 at each node k repeat operation:

$$\begin{cases} \boldsymbol{\psi_{k.i} = w_{k.i-1} + \mu_k \sum_{l \epsilon N_k} c_{k.i} u_{l,i}^* \left(d_l(i) - u_{l,i} w_{k.i-1}\right)} & \quad \boldsymbol{incremental\ step} \\ \quad \boldsymbol{w_{k.i-1} = \sum_{l \epsilon N_k} a_{l,k} \psi_{l.i}} & \quad \boldsymbol{diffusion\ step} \end{cases} \qquad \mathbf{5\text{-}15}$$

The ATC algorithm is followed by incremental diffusion update $w_{k.i-1} = \sum_{l \epsilon N_k} a_{l,k} \psi_{l.i}$. the coefficient $c_{k.i}$ determine node in vicinity of $l \epsilon N_k$ sharing measurements $\{d_l(i), u_{l,i}\}$ With corresponding running node k. while for diffusion step for deciding node in $l\epsilon N_k$sending intermediate estimation $\{\psi_{l.i}\}$ with node k . This is a convex combinational approach than the form that employed before adaptive filtering [29].

If the above mentioned process is reversed for incremental updating combine-then-adapt diffusion LMS algorithm is formulated. Star with $\{w_{l,-1} = 0\}$ for all l .for real non-negative real coefficients$\{c_{l,k}, a_{l,k}\}$ for each time i≥0 at each node k repeat operation:

$$\begin{cases} \boldsymbol{\psi_{k.i-1}} = \sum_{l\epsilon N_k^*} \boldsymbol{a_{l,k} w_{l.i-1}} & \boldsymbol{diffusion\ step} \\ \boldsymbol{w_{k.i}} = \boldsymbol{\psi_{k.i-1}} + \boldsymbol{\mu_k} \sum_{l\epsilon N_k} \boldsymbol{c_{l,k} u_{l,i}^*} \left(\boldsymbol{d_l(i)} - \boldsymbol{u_{l,i} \psi_{k.i-1}}\right) & \boldsymbol{incremental\ step} \end{cases} \quad \textbf{5-16}$$



(A)ATC diffusion strategy

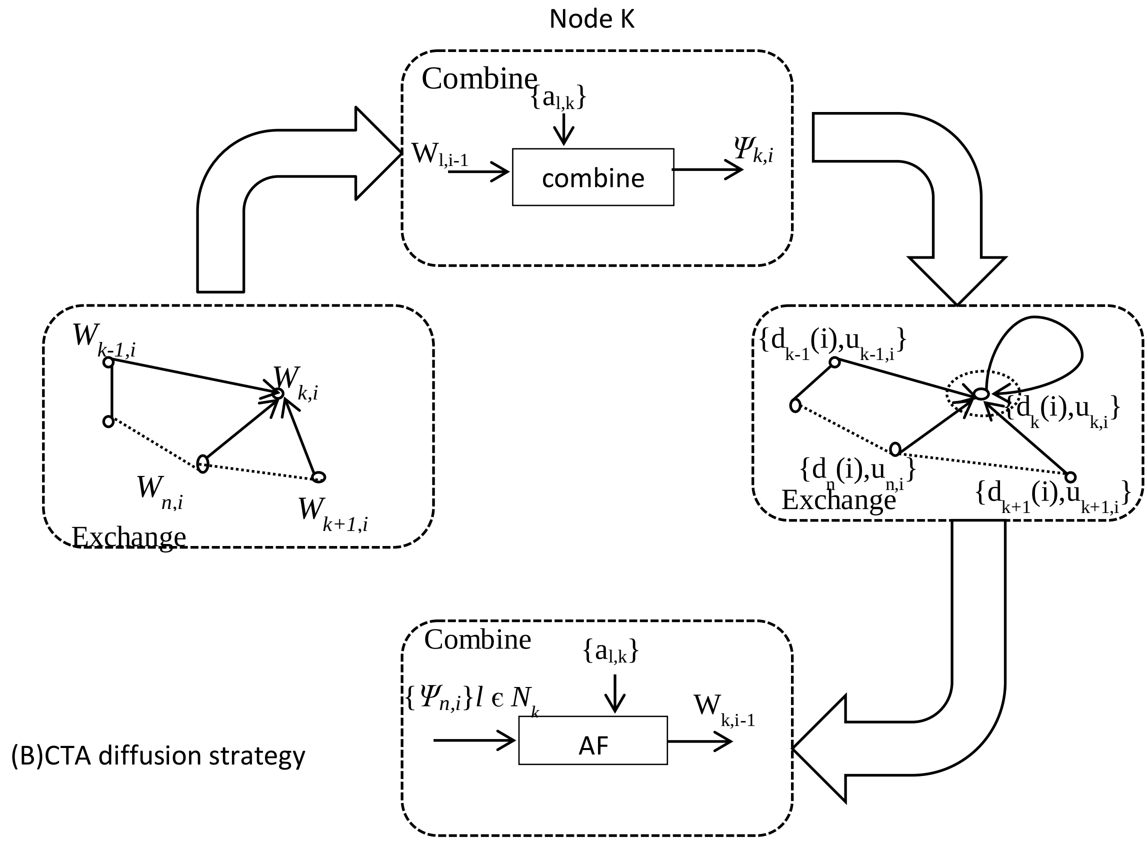


Figure 5-1 Diffusion strategy (a)ATC (b)CTA

5.5 Network Global Model

The algorithm (5.12) comprises of interconnected adaptive filter updates. The global quantities required are

$$\boldsymbol{\psi}^{\mathbf{j}} = \left[\boldsymbol{\psi}_1^{\mathbf{j}}\ \boldsymbol{\psi}_2^{\mathbf{j}}\ \boldsymbol{\psi}_3^{\mathbf{j}}\ \boldsymbol{\psi}_4^{\mathbf{j}} \dots\dots\dots\dots.\ \boldsymbol{\psi}_{\mathbf{N}}^{\mathbf{j}}\right]^{\mathbf{T}} \quad , \quad \boldsymbol{\phi}^{\mathbf{j}} = \left[\boldsymbol{\phi}_1^{\mathbf{j}}\ \boldsymbol{\phi}_2^{\mathbf{j}}\ \boldsymbol{\phi}_3^{\mathbf{j}}\ \boldsymbol{\phi}_4^{\mathbf{j}} \dots\dots\dots\dots.\ \boldsymbol{\phi}_{\mathbf{N}}^{\mathbf{j}}\right]^{\mathbf{T}}$$

$$\mathbf{U_j} = \mathbf{diag}\left\{\mathbf{u_{1,j}}\ \mathbf{u_{2,j}} \dots..\ \mathbf{u_{N,j}}\right\} \quad , \quad \mathbf{d_j} = \mathbf{diag}\left\{\mathbf{d_{1,j}}\ \mathbf{d_{2,j}} \dots..\ \mathbf{d_{N,j}}\right\}$$

$$\mathbf{D} = \mathbf{diag}\{\boldsymbol{\mu}_1\mathbf{I_M}\ \boldsymbol{\mu}_2\mathbf{I_M}\ \boldsymbol{\mu}_3\mathbf{I_M} \dots\dots\ \boldsymbol{\mu}_{\mathbf{N}}\mathbf{I_M}\} \quad (\mathbf{NM} \times \mathbf{NM}) \qquad \mathbf{5\text{-}17}$$

The D matrix is called as a diagonal matrix configuring local step sizes. Measurements are configured from by traditional modeling of form

$$\mathbf{d_k}(\mathbf{j}) = \mathbf{u_{k,j}}\mathbf{w^0} + \mathbf{v_k}(\mathbf{j}) \qquad \mathbf{5\text{-}18}$$

$v_k(j)$ is background noise assuming independent over time and with variance $\sigma^2_{v,k}$. Linear models of the form (5.17) are able to capture or approximate many input output relations for estimation purposes. The global liner model be

$$\mathbf{d_j = U_j w^{(0)} + v_j} \qquad \textbf{5-19}$$

Where $w^{(0)} = Qw^0$ and $v_j = [v_{1,j}\ \ v_{2,j}\ \ \ldots\ldots\ \ v_{N,j}] \quad (N \times 1)$

With the algorithm (5.13)as in global form

$$\begin{aligned}\boldsymbol{\phi}^{\mathbf{j-1}} &= \mathbf{G}\boldsymbol{\psi}^{\mathbf{j-1}}\\ \boldsymbol{\psi}^{\mathbf{j}} &= \boldsymbol{\phi}^{\mathbf{j-1}} + \mathbf{DU_j^*}\left(\mathbf{d_j} - \mathbf{U_j}\boldsymbol{\phi}^{\mathbf{j-1}}\right)\end{aligned} \qquad \textbf{5-20}$$

$$\boldsymbol{\psi}^{\mathbf{j}} = \mathbf{G}\boldsymbol{\psi}^{\mathbf{j-1}} + \mathbf{DU_j^*}\left(\mathbf{d_j} - \mathbf{U_j G}\boldsymbol{\psi}^{\mathbf{j-1}}\right) \qquad \textbf{5-21}$$

Where $G = C \otimes I_M$is NM×NM transition matrix and C is N×N diffusion combination matrix with coefficient as correlation matrix [ckl].

Combiner coefficient C is$C_{qN} = \mathrm{col}\{1,1, \ldots 1\}$. This combiner matrix C can be Metropolis, Laplacian, nearest neighbor rule [30] [31]. All these rules are described as below

If n_k and n_l denote the degree for nodes k and l, I.e. $n_k = |N_k|$. The metropolis rule is

$$\mathbf{c_{kl}} = \begin{cases} \frac{\mathbf{1}}{\mathbf{max(n_k, n_l)}} & \textbf{if } \mathbf{k \neq l} \textbf{ are linked} \\ \mathbf{0} \quad \textbf{for kand l are not linked} \\ \quad \mathbf{1 - \sum_{l \epsilon N_k/k} c_{kl}} \textbf{for } \mathbf{k = l} \end{cases} \qquad \textbf{5-22}$$

The laplacian rule is

$$C = I_N - \mathcal{K}\mathcal{L}$$

$\mathcal{L} = \mathcal{D} - \mathcal{A}_d$ with $\mathcal{D} = \mathrm{diag}\{n_1\ n_2\ n_3\ \ldots.\,n_N\}, \mathcal{K} = \frac{1}{n_{max}}$ and Ad is N×N network adjacent matrix form as

$$[\mathbf{A_d}]_{\mathbf{kl}} = \begin{cases} \mathbf{1}, \textbf{if k and l are linked} \\ \quad \mathbf{0} \textbf{ otherwise} \end{cases} \qquad \textbf{5-23}$$

For the nearest neighbour rule, the combiner matrix C is defined as

$$\mathbf{c_{kl}} = \begin{cases} \frac{\mathbf{1}}{\mathbf{N_k}}, & \mathbf{l \in N_k} \\ \mathbf{0} & \mathbf{otherwise} \end{cases} \quad \mathbf{5\text{-}24}$$

Performance Analysis

The performance analysis in an interconnected network is a challenging job due to the following reasons:

1. Each node k is influenced by the local data with local statistics$\{R_{yx,k}\ R_{x,k}\}$.
2. Each node k is influenced by its neighborhood nodes through local diffusion.
3. Each node is influenced by the local noise with variance$\sigma^2_{v,k}$.

The energy based approach is extended to the space dimension [30] because the distributed adaptive algorithm (5.14) involves both the time variable j(block number) and space variable k. We will define the common terms MSD (Mean Square Deviation), MSE (Mean Square Error) and EMSE (Excess Mean Square Error) for local and also for global network [32].

5.5.1 Mean transient analysis

The global weight error vector

$$\bar{\boldsymbol{\psi}}^{\mathbf{j}} = \mathbf{w}^{(\mathbf{0})} - \boldsymbol{\psi}^{\mathbf{j}} \quad \mathbf{5\text{-}25}$$

As $\mathrm{Gw}^{(0)} = \mathrm{w}^{(0)}$using the global data (5.17) and subtracting w0 from the left hand side and Gw0 from the right side of (5.19),

$$\bar{\boldsymbol{\psi}}^{\mathbf{j}} = \mathbf{G}\bar{\boldsymbol{\psi}}^{\mathbf{j-1}} - \mathbf{D}\mathbf{U}_{\mathbf{j}}^{*}\left(\mathbf{U}_{\mathbf{j}}\mathbf{w}^{(\mathbf{0})} + \mathbf{v}_{\mathbf{j}} - \mathbf{U}_{\mathbf{j}}\mathbf{G}\bar{\boldsymbol{\psi}}^{\mathbf{j-1}}\right) \quad \mathbf{5\text{-}26}$$

$$\bar{\boldsymbol{\psi}}^{\mathbf{j}} = \left(\mathbf{I_{NM}} - \mathbf{D}\mathbf{U}_{\mathbf{j}}^{*}\mathbf{U}_{\mathbf{j}}\right)\mathbf{G}\bar{\boldsymbol{\psi}}^{\mathbf{j-1}} - \mathbf{D}\mathbf{U}_{\mathbf{j}}^{*}\mathbf{v}_{\mathbf{j}} \quad \mathbf{5\text{-}27}$$

Assuming temporal and spatial independence of the regression data $\{u_{k,j}\}$ and taking the expectations of both sides of the above equation (5.26) leads to

$$\mathbf{E}[\bar{\boldsymbol{\psi}}^{\mathbf{j}}] = (\mathbf{I_{NM}} - \mathbf{D}\mathbf{R_u})\mathbf{G}\mathbf{E}[\bar{\boldsymbol{\psi}}^{\mathbf{j-1}}] \quad \mathbf{5\text{-}28}$$

Where $R_u = \mathrm{diag}\{R_{u,1}\ R_{u,2} \ldots .\ R_{u,N}\}$ is block diagonal vector and $R_{u,k} = E\left[u^*_{k,j}u_{k,j}\right]$. In the absence of cooperation (i.e., when the nodes evolve independently of each other and therefore

$G = I_{NM}$, the mean error vector would evolve according to

$$\mathbf{E}[\bar{\boldsymbol{\psi}}^{\mathbf{j}}] = (\mathbf{I}_{\mathbf{NM}} - \mathbf{DR}_{\mathbf{u}})\mathbf{E}[\bar{\boldsymbol{\psi}}^{\mathbf{j-1}}] \qquad \textbf{5-29}$$

5.5.2 Mean Square Transient Analysis

To perform the transient analysis of the adaptive network and characterize the evolution of its learning curves .Here for deriving the expressions for mean-square deviation (MSD) and excess-mean-square-deviation(EMSE).

The local output estimation error at node k is

$$\mathbf{e}_{\mathbf{k}}^{\mathbf{j}} = \mathbf{d}_{\mathbf{k}}^{\mathbf{j}} - \mathbf{u}_{\mathbf{k,j}}\boldsymbol{\phi}_{\mathbf{k}}^{\mathbf{j-1}} \qquad \textbf{5-30}$$

For global error vector across network $e_j = \left[e_{1,j}\ e_{2,j}\ \ldots .\ e_{N,j}\right]^T \quad (N \times 1)$

$$\mathbf{e}_{\mathbf{j}} = \mathbf{d}_{\mathbf{j}} - \mathbf{U}_{\mathbf{j}}\mathbf{G}\bar{\boldsymbol{\Psi}}^{(\mathbf{j-1})} = \mathbf{U}_{\mathbf{j}}\mathbf{G}\bar{\boldsymbol{\Psi}}^{(\mathbf{j-1})} + \mathbf{v}_{\mathbf{j}} = \mathbf{e}_{\mathbf{a,j}}^{\mathbf{G}} + \mathbf{v}_{\mathbf{j}} \qquad \textbf{5-31}$$

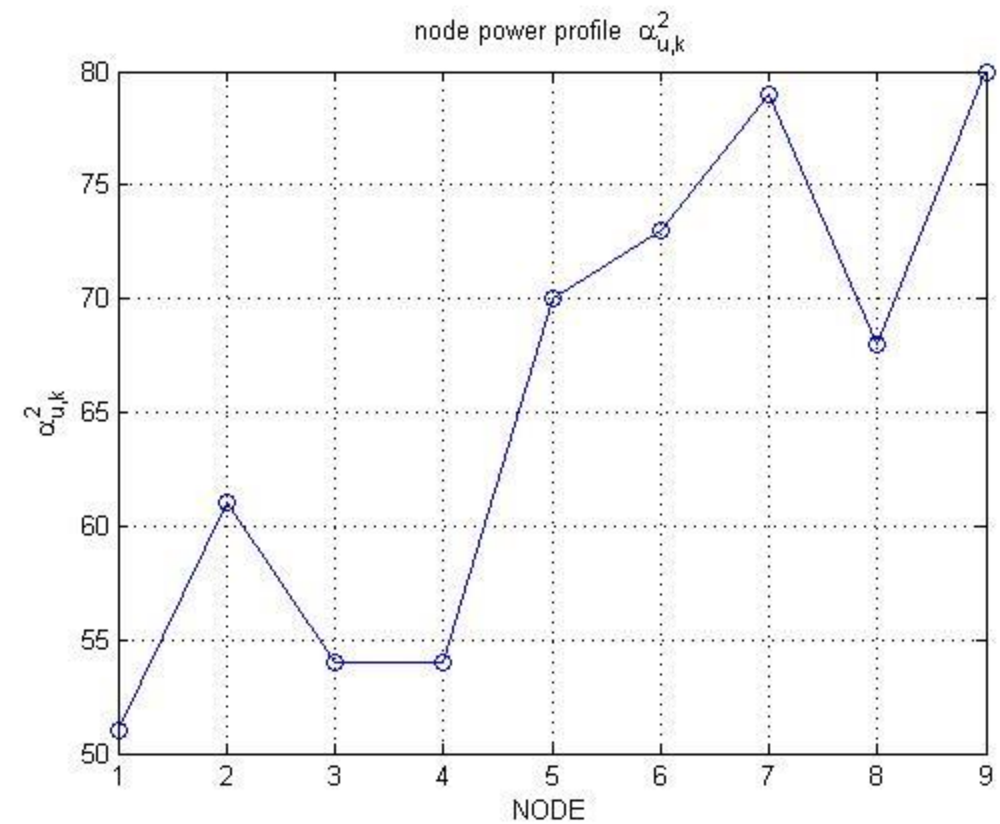
Where $e_{a,j}^{G} = U_j G \bar{\Psi}^{(j-1)}$

For definition of a priori and a posteriori weighted estimation error:

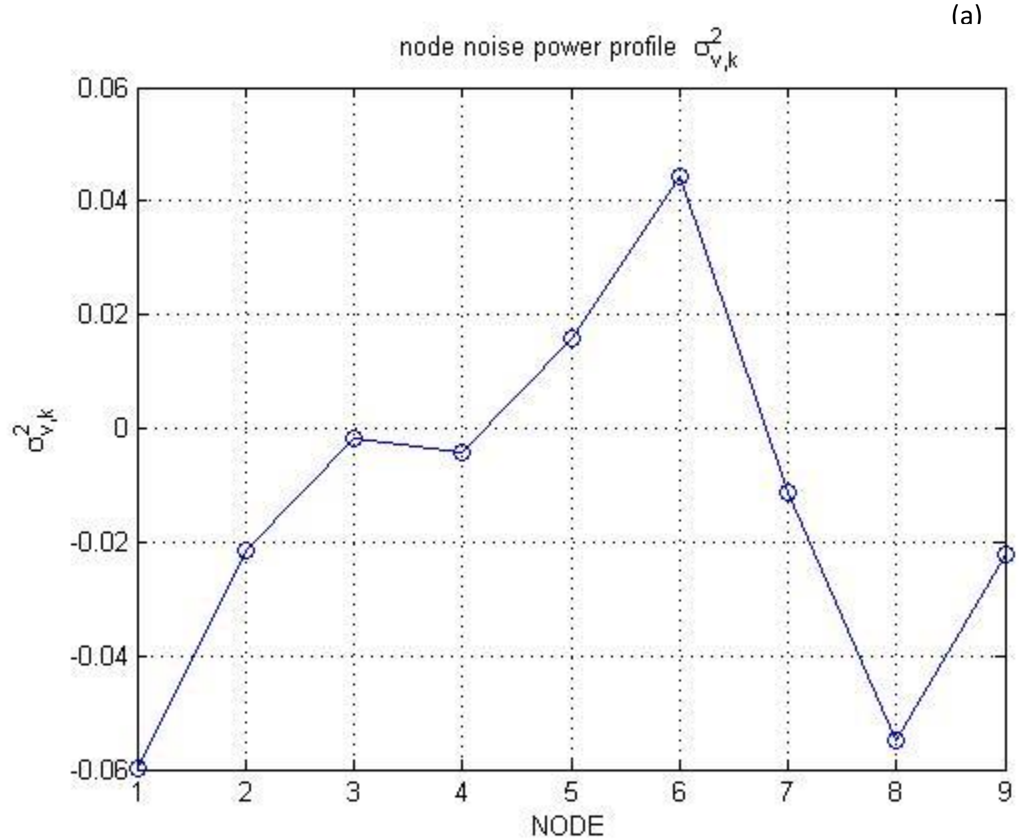
$$\mathbf{e}_{\mathbf{a,j}}^{\mathbf{D}\sum\mathbf{G}} = \mathbf{U}_{\mathbf{j}}\mathbf{D}\sum\mathbf{G}\bar{\boldsymbol{\psi}}^{\mathbf{j-1}} \quad \textbf{and} \quad \mathbf{e}_{\mathbf{p,j}}^{\mathbf{D}\sum\mathbf{G}} = \mathbf{U}_{\mathbf{j}}\mathbf{D}\sum\mathbf{G}\bar{\boldsymbol{\psi}}^{\mathbf{j}} \qquad \textbf{5-32}$$

5.6 Simulation Result Discussion

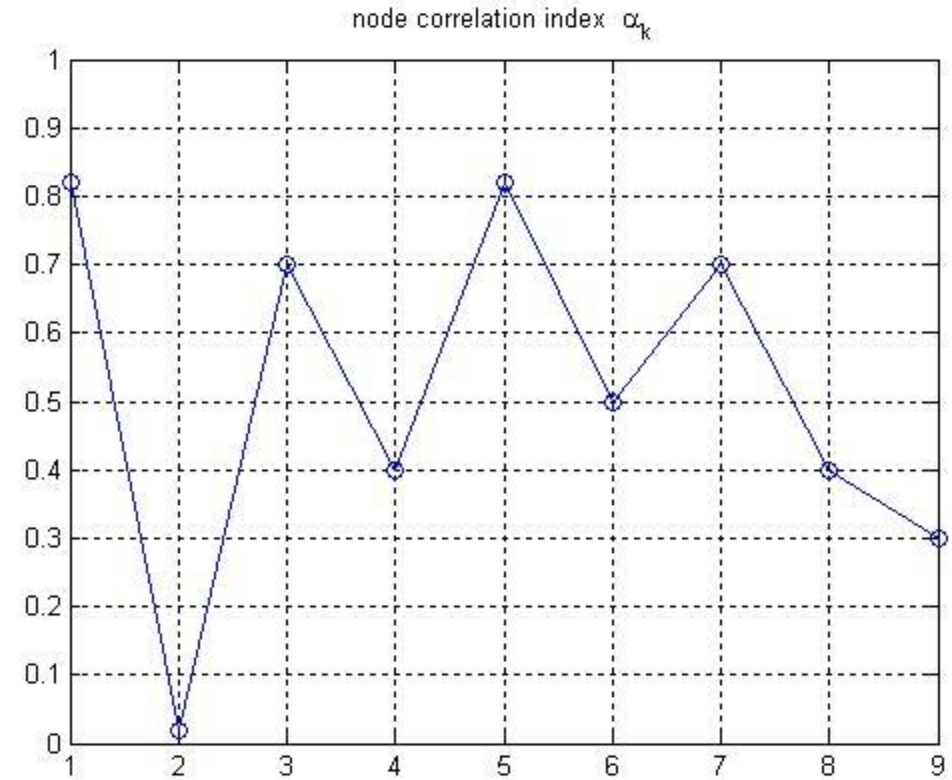
The simulation result is produced for node N=9. The back ground noise power = 10-3. Network statistics are given as under. There are 20 independent experiments are conducted and various error is averaged for display purposes. The error curves are generated from learning process for 2000 iteration. The error are collected over samples and averaged over number of experiments. The global mean square deviation MSD is calculated from averaging $E\left\|\bar{\psi}_k^{j-1}\right\|$ across nodes over 20 experiments. The global excess mean square error EMSE $E\left\|e_{a,k}(j)\right\|^2$ where $e_{a,k}(j) = x_{k,j}\bar{\psi}_k^{j-1}$ mean square error MSE $E\left\|d_k - x_{k,j}\bar{\psi}_k^{j-1}\right\|$ is also conducted as described for MSD. For local error estimation of MSE, EMSE,MSD at node 1.


node power profile $\alpha^2_{u,k}$
80
75
70
65
60
55
50
$\alpha^2_{u,k}$
1
2
3
4
5
6
7
8
9
NODE

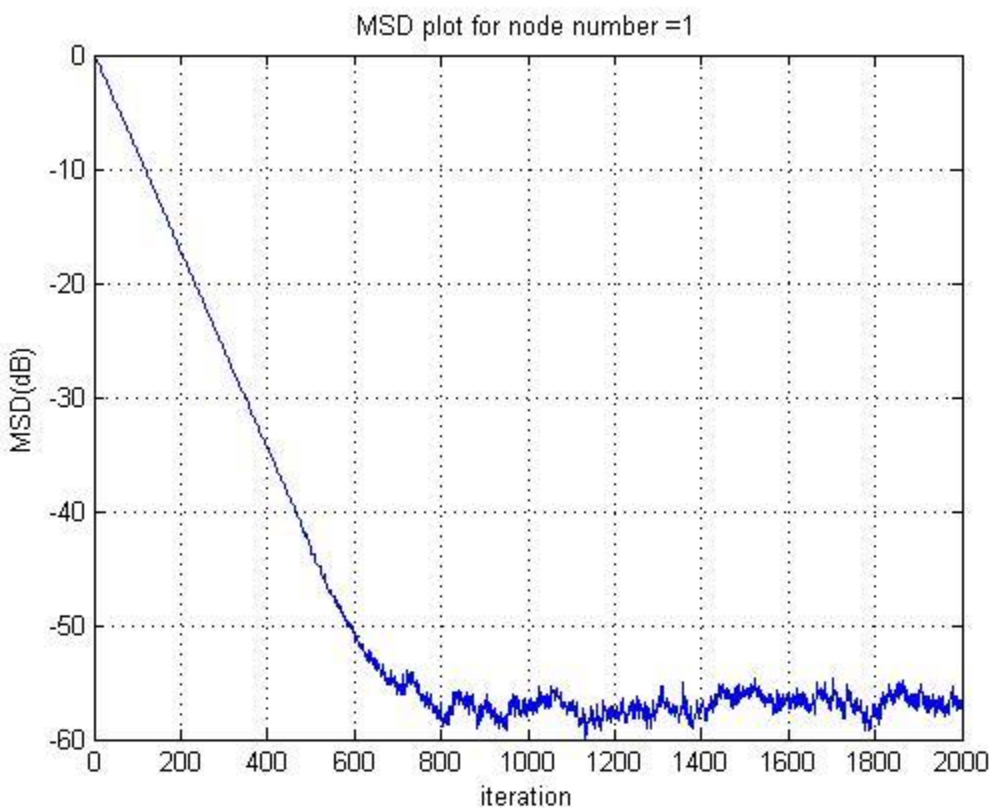
(a)


node noise power profile $\sigma^2_{v,k}$
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
$\sigma^2_{v,k}$
1
2
3
4
5
6
7
8
9
NODE

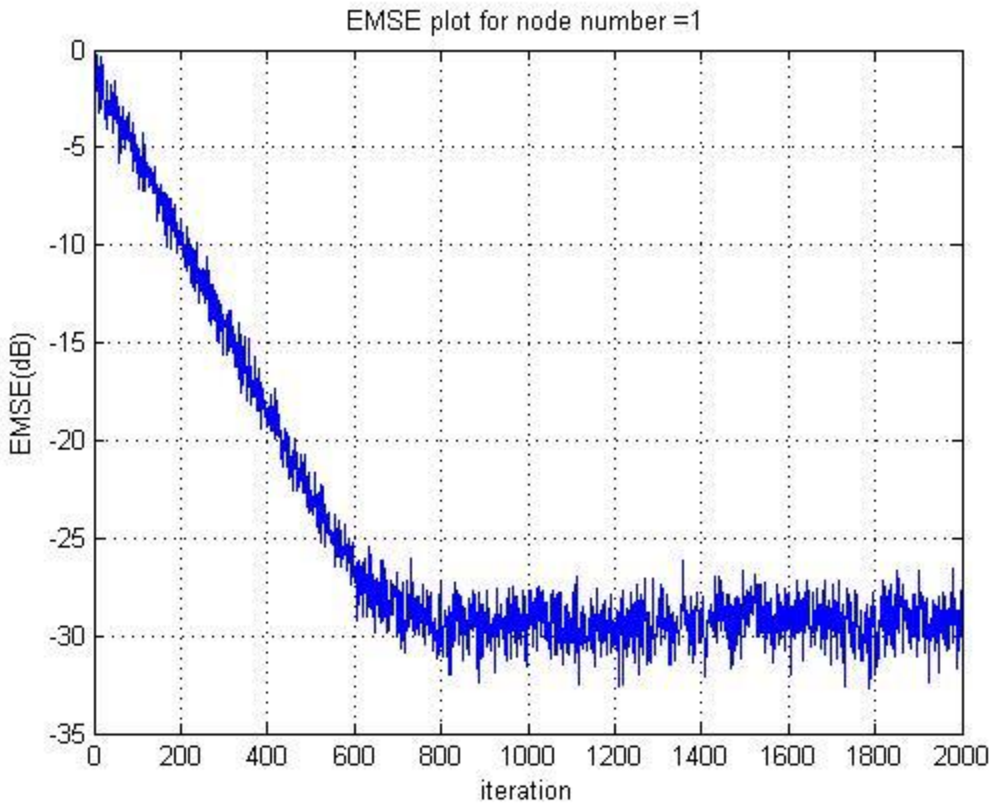
(b)


node correlation index α_k
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2
3
4
5
6
7
8
9

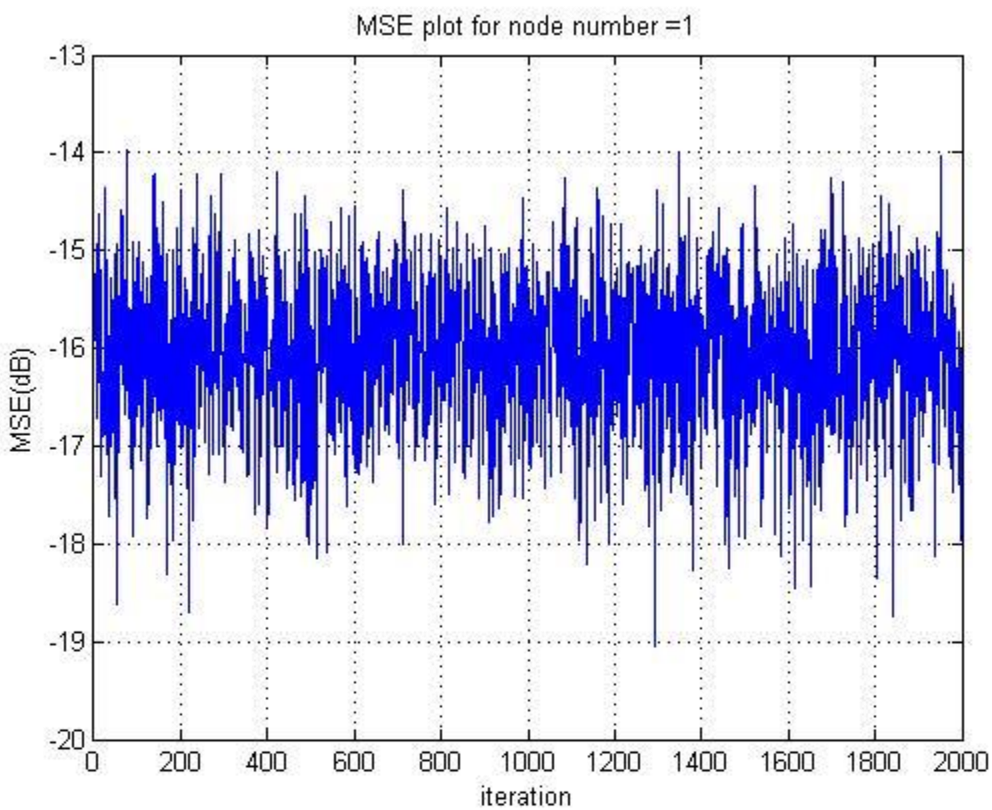
(c)


MSD plot for node number =1
MSD(dB)
0
-10
-20
-30
-40
-50
-60
0
200
400
600
800
1000
1200
1400
1600
1800
2000
iteration

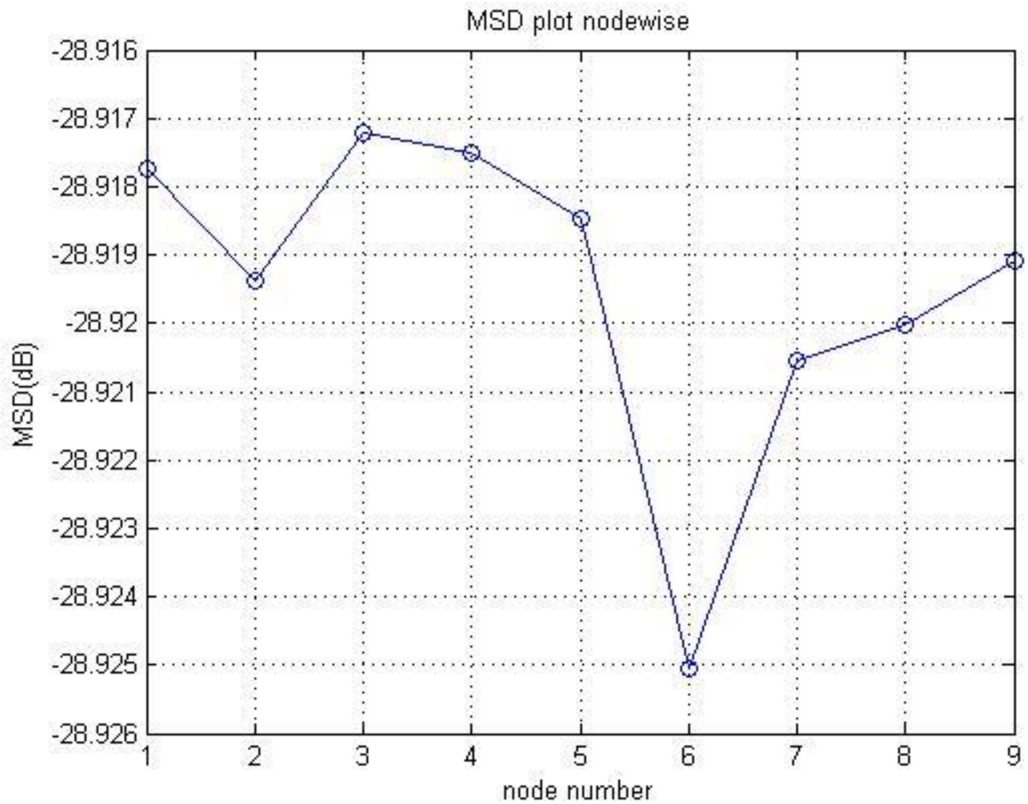
(d)


EMSE plot for node number =1
0
-5
-10
-15
-20
-25
-30
-35
EMSE(dB)
0
200
400
600
800
1000
1200
1400
1600
1800
2000
iteration

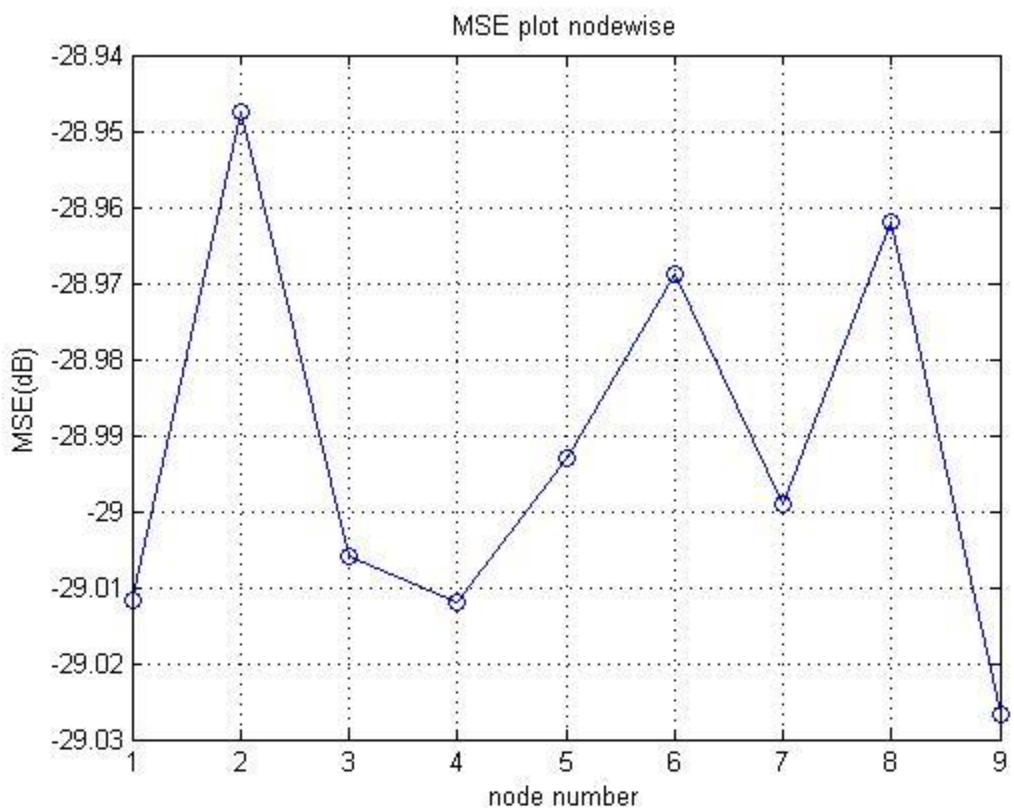
(e)


MSE plot for node number =1
-13
-14
-15
-16
-17
-18
-19
-20
MSE(dB)
0
200
400
600
800
1000
1200
1400
1600
1800
2000
iteration

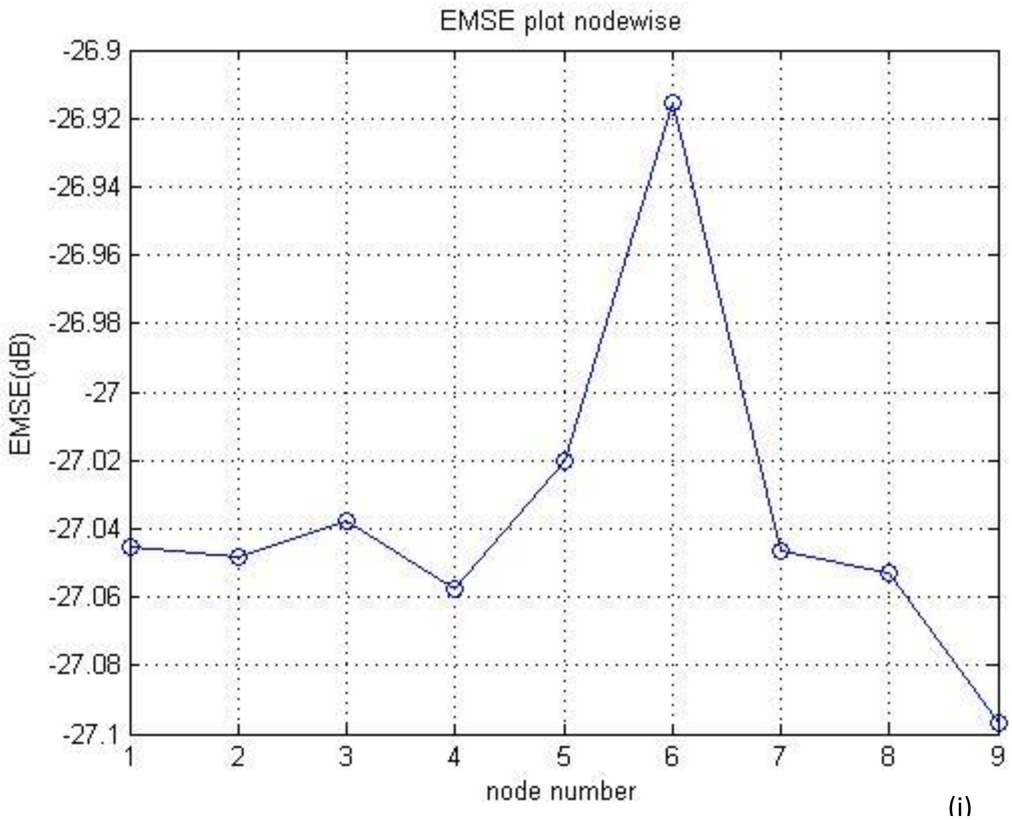
(f)


MSD plot nodewise
MSD(dB)
-28.916
-28.917
-28.918
-28.919
-28.92
-28.921
-28.922
-28.923
-28.924
-28.925
-28.926
1
2
3
4
5
6
7
8
9
node number

(g)


MSE plot nodewise
MSE(dB)
-28.94
-28.95
-28.96
-28.97
-28.98
-28.99
-29
-29.01
-29.02
-29.03
1
2
3
4
5
6
7
8
9
node number

(h)



(i)

Figure 5-2 cooperative diffusion strategy for N=9 (a) node power profile (b) noise power profile (c) correlation index (d) MSD at node 1 (e) EMSE at node 1 (f) MSE at node 1 (g) MSD node wise (h) MSE node wise (i) EMSE node wise

5.7 Partial update in diffusion strategy

As per partial update technique as discussed in chapter 4 for incremental adaptive strategy here the behavior for the diffusion algorithm is studied. As per experiment setup totally 9 node are taken. To each node there is data stream of 8100 samples. There are 20 uncorrelated experiments are carried out. The back ground noise power = 10^{-3} keeping SNR ration to be -30 dB around node. The error are collected over samples and averaged over number of experiments.

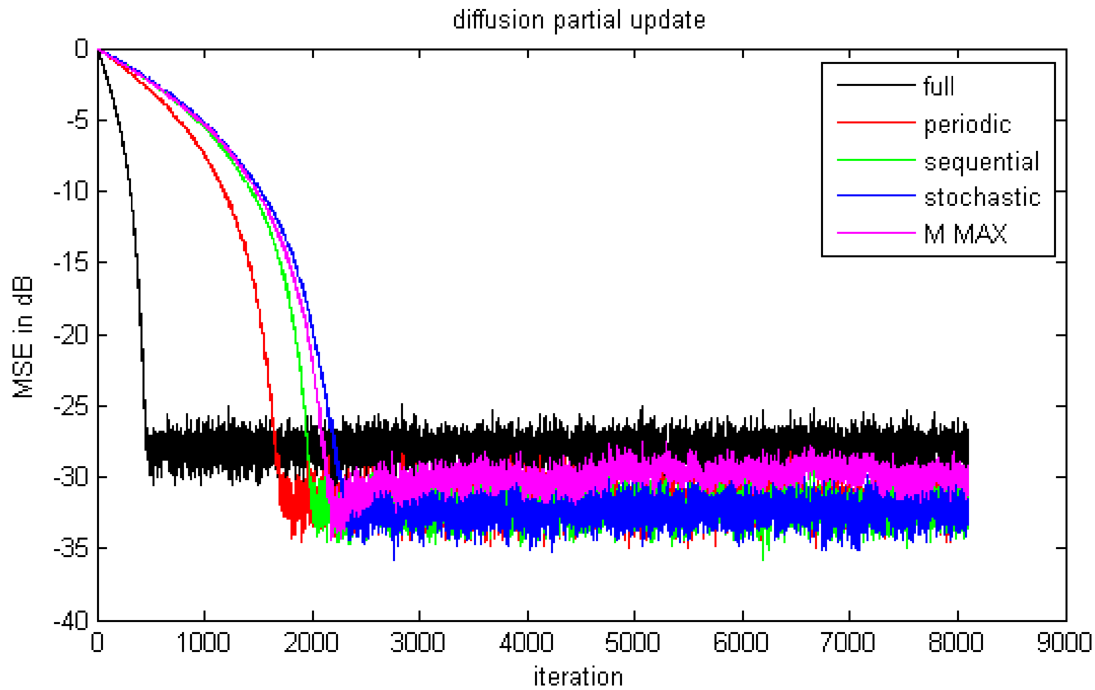
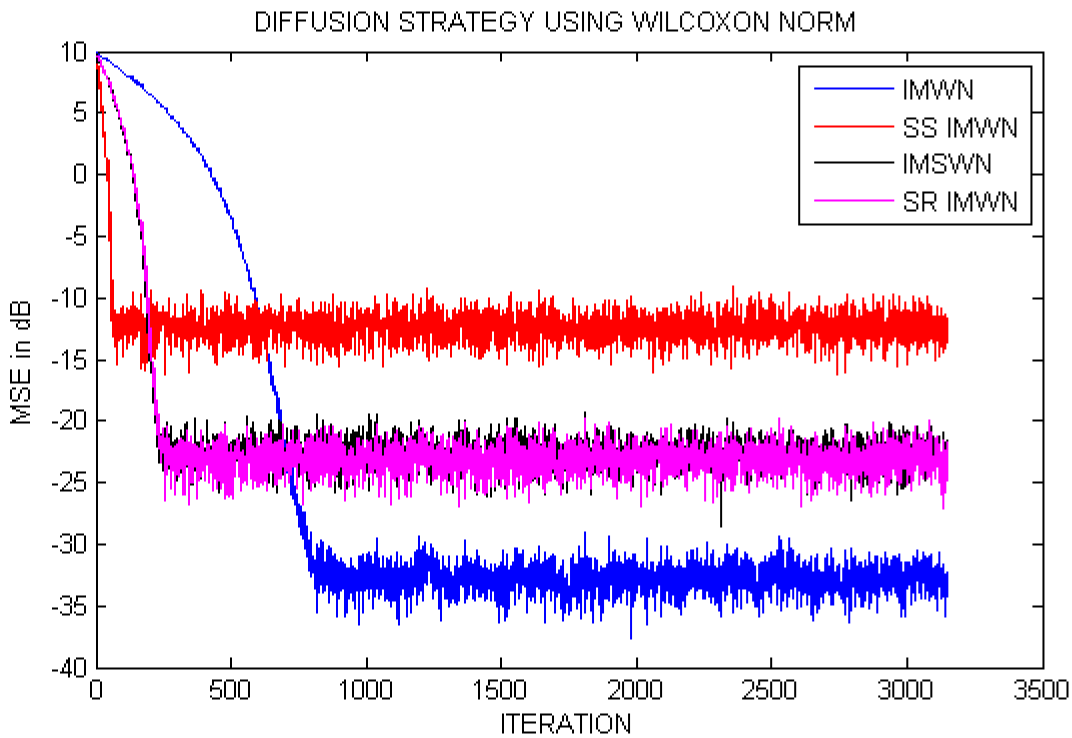


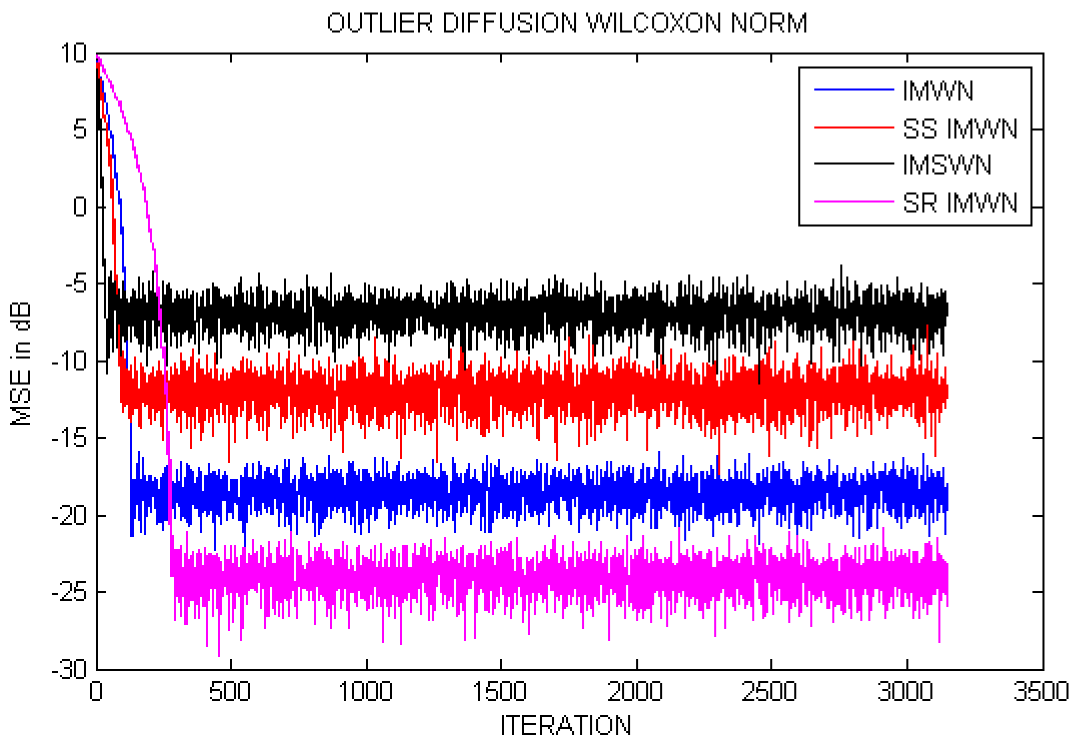
Figure 5-3 partial update using diffusion strategy M=10 ;N=4

5.8 Regression Analysis with Wilcoxon Norm

In the statistical regression technique as discussed in chapter 3.4 for incremental adaptive strategy here the behavior for the diffusion algorithm is studied. As per experiment setup totally 9 node are taken. To each node there is data stream of 3200 samples. There are 20 uncorrelated experiments are carried out. The back ground noise power = 10^{-3} keeping SNR ration to be -30 dB around node. The error are collected over samples and averaged over number of experiments. For impulsive noise of 30% is also tested in next figure.



(a)



(b)

Figure 5-4 Regression analysis (a) 0% impulse noise (b) 30% impulse noise

5.9 Conclusion

The mathematical analysis and the simulation results show that cooperation improves performance by reducing computation and communication resources. It has a stabilizing effect on the distributed network. Filters can be designed using local information to achieve local stability and diffusion protocols can be implemented to improve the global performance. Closed-form expressions for global and local mean and mean-square performance have been derived, matching very well the simulations that have been carried out. Distributed LMS algorithm in which the block concept has been incorporated into the diffusion LMS algorithm discussed in. The mathematical analysis and the simulation results show that the performance of the diffusion block LMS algorithm is nearly same as diffusion LMS. The number of communications between neighbor nodes decrease to frac1L times the number of communications in diffusion LMS. This approach is preferred in those applications where there is a severe bound on communication resources.

Chapter 6

Conclusion and Future Work

6. Chapter 6::Conclusion and Future Work

Several efforts have been pursued in the literature to develop distributed estimation schemes based on consensus strategies. One of the main results of this work is to show that cooperation improves performance from the estimation point of view, in terms of saving computation and communication resources. Cooperation has a stabilizing effect on the network. One can design the individual filters using local information only in order to achieve (local) stability and implement incremental or diffusion protocols to improve global performance. Energy conservation arguments have been used to study the steady state performance of the individual nodes for Gaussian data. Closed-form expressions for global and local mean and mean-square performance have been derived, matching very well the simulations that have been carried out. The inherent cooperative strategy of the incremental scheme not only improves performance, but it also decreases the amount of communication needed to implement cooperation among the nodes. The diffusion scheme results in peer-to-peer algorithms suitable for general topologies and robust to link and node failures. Besides robustness and spatial diversity, diffusion protocols improve the network estimation performance but with an additional level of complexity.

In this thesis we have studied the deployment problem for mobile wireless sensor networks. The limitation is for a certain number of nodes with limited sensing and communication range. The scenario consists of a" random" distribution of nodes over the region of interest. Though many scenarios adopt random deployment because of practical reasons such as deployment cost and time, random deployment may not provide a uniform distribution which is desirable for a longer system lifetime over the region of interest. In this thesis, we have proposed a multi-objective approach for the deployment of nodes to improve upon an irregular initial deployment of nodes. Coverage and lifetime are taken as the two conflicting objectives for achieving a set of layouts.

6.1 Scope for Future Work

The study of convergence analysis in error surface is to be analyzed. Various procedures for convergence plane can be applied for better result estimation. The quantization effect can be a reduced by "offset truncation rule", "error feedback quantizer", "memory size expansion method" and "leakage algorithm". Kernel adaptive filtering is an adaptive filtering technique for general nonlinear problems. It is a natural generalization of linear adaptive filtering in reproducing kernel Hilbert spaces. A covariance matrix estimation scheme takes into account the possible presence of outliers, without censoring any training sample. A Bayesian model is formulated where the amplitude of the signal component of each training sample is assumed to follow a Bernoulli-Gaussian distribution. The conditional-error-covariance matrix of the estimator is also obtained in a form suitable for on-line performance evaluation.

7. Bibliography

[1] A. H. Sayed, Fundamentals of Adaptive Filtering, New York: Wiley, 2003.

[2] L. Ljung, "Perspectives on system identification," *Elsevier Science Ltd.,* vol. 34, no. 1, pp. 1-12, April 2010.

[3] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control,* Vols. AC-19, no. 6, pp. 716-23, Dec. 1974.

[4] L. L., System Identification, Englewood Cliffs: Prentice-Hall, 1987.

[5] D. S. S. Widrow Bernerd, Adaptive Signal Processing, london: Pearson Education, 2002.

[6] W. N., Extrapolation, Interpolation and smoothing of time series with engineering application, Cambridge: MIT Press, 1949.

[7] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine,* vol. 40, no. 8, pp. 102-105, August 2002.

[8] D. Estrin, R. Govindan, J. Heidemann and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Proceedings of 5th Annual Joint ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'99)*, Seattle, WA, USA, 1999.

[9] M. Wax and T. Kailath, "Decentralized processing in sensor arrays," *IEEE Transactions on Acoustics, Speech and Signal Processing,* Vols. ASSP-33, no. 5, pp. 1123-9, Oct. 1985.

[10] "Detection, classification, and tracking of targets," *IEEE Signal Processing Magazine,* vol. 19, no. 2, pp. 17-29, March 2002.

[11] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine,* vol. 40, no. 8, pp. 102-14, Aug. 2002.

[12] D. E. a. M. S. D. Culler, "Overview of Sensor Networks," *IEEE Computer Society,* vol. 37, no. 8, pp. 41 - 49 , Aug. 2004 .

[13] A. Elfes, "Sonar based real world mapping and navigation," *Robotics and Automation, IEEE Journal,* vol. 3, no. 3, pp. 249 - 265 , June 1987 .

[14] C. G. Lopes and A. H. Sayed, "Incremental Adaptive Strategies Over Distributed Networks," *IEEE Transactions on Signal Processing,* vol. 55, no. 8, pp. 4064-4077, August 2007.

[15] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis," in *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2006*, Toulouse, France, 2006.

[16] D. Estrin, R. Govindan, J. Heidemann and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Proceedings of 5th Annual Joint ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'99)*, Seattle, WA, USA, 1999.

[17] S. Dhillon, K. Chakrabarty and S. Iyengar, "Sensor placement for grid coverage under imprecise detections," *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997),* vol. 2, no. 1, pp. 1581-7, 2002.

[18] J. McKean, J. Naranjo and S. Sheather, " Diagnostics to detect differences in robust fits of linear models," *Comput. Stat. (Germany),* vol. 11, no. 3, pp. 223-43, 1996.

[19] H. Leung and S. Haykin, "Detection and estimation using an adaptive rational function filter," *IEEE Transactions on Signal Processing,* vol. 42, no. 12, pp. 3366-3376, Dec 1994.

[20] K. Dogancay, partial update adaptive filter and adaptive signal processing, 11 Nov 2008.

[21] S. Douglas, "Simplified stochastic gradient adaptive filters using partial updating," in *Proceedings of IEEE 6th Digital Signal Processing Workshop*, New York, NY, USA, 1994.

[22] R. Han and D. Messerschmitt, "Progressively reliable transport protocol for interactive wireless multimedia," *Multimedia systems ,* vol. 7, no. 2, pp. 141-156, March 1999.

[23] M. Godavarti and A. Hero, "Stochastic partial update LMS algorithm for adaptive arrays," in *Proceedings of the 2000 IEEE Sensor Array and Multichannel Signal Processing Workshop*, Piscataway, NJ, USA, 16-17 March 2000.

[24] K. Mayyas, T. Aboulnasr and T. Eldos, "A study of the robustness of the M-Max NLMS adaptive algorithm," in *ISCAS'99. Proceedings of the 1999 IEEE International Symposium on Circuits and Systems. VLSI*, Piscataway, NJ, USA, 1999.

[25] B. S. Lin Xiao, "Fast linear iterations for distributed averaging," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference*, Stanford Univ., CA, USA, Dec. 2003.

[26] A. Jadbabaie, J. Lin and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *Automatic Control, IEEE Transactions,* vol. 46, no. 6, pp. 988 - 1001, June 2003.

[27] E. S. a. A. H. S. C. G. Lopes, "Adaptive Carrier Tracking for Direct-To-Earth Mars Communications," in *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference*, Pacific Grove, CA,

Oct. 29 2006-Nov. 1 2006 .

[28] C. Lopes, E. Satorius, P. Estabrook and A. Sayed, "Efficient Adaptive Carrier Tracking for Mars to Earth Communications During Entry, Descent and Landing," in *Statistical Signal Processing, 2007. SSP '07. IEEE/SP 14th Workshop*, Madison, WI, USA , 26-29 Aug. 2007.

[29] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS Strategies for Distributed Estimation," *IEEE TRANSACTIONS ON SIGNAL PROCESSING,* vol. 58, no. 3, pp. 1035 - 1048 , MARCH 2010.

[30] C. G. Lopes and A. H. Sayed, "Diffusion Least-Mean Squares Over Adaptive Networks: Formulation and Performance Analysis," *Signal Processing, IEEE Transactions,* vol. 56, no. 7, pp. 3122 - 3136 , July 2008.

[31] B. K. a. C. J. K. L. A. Rossi, "Distributed parameter estimation for monitoring diffusion phenomena using physical models," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference*, Los Angeles, CA, USA, Oct. 2004 .

[32] J. Arenas-Garcia, A. Figueiras-Vidal and A. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Signal Processing Society,* vol. 54, no. 3, pp. 1078 - 1090, March 2006.